Grado en Ingeniería Informática

Trabajo Final de Grado

Aplicación de gestión de playas

Autor:
Daniel Higueras Goold

Supervisor:
Héctor Segarra Castillo
Tutor académico:
José Luis Llopis Borrás

Fecha de lectura: 14 de Noviembre de 2014
Curso académico 2013/2014

Agradecimientos

Para la realización de la memoria quisiera agradecer el apoyo que he recibido por parte de mi familia y amigos. También quisiera agradecer la ayuda recibida por parte de los trabajadores de la empresa en la que he realizado las prácticas, así como al tutor y al supervisor de las mismas. Para finalizar, quisiera agradecer el apoyo económico recibido por parte del Banco Santander, mediante la concesión de la beca CRUE CEPYME 2014.

Resumen

La empresa Inttegrum Consultora y Gestora del Mediterráneo S.L. es una empresa dedicada al desarrollo, mantenimiento y venta de diferentes soluciones software y hardware. Su cliente Mediterránea Grupo vino con un problema; necesitaba mejorar sus métodos de gestión del socorrismo, meteorología, incidentes, salidas de embarcaciones y el material prestado de las playas que tenía asignadas. También debía servir como ventaja a la hora de concursar por las playas en los ayuntamientos.

De esto surge este proyecto, en el cual se han realizado todas las fases del desarrollo de una aplicación de gestión de estos aspectos. Para ello se ha seguido una metodología tradicional en cascada con apoyo de una herramienta de gestión de tareas. Ha sido implementado utilizando una base de datos MySql, el lenguaje PHP, y el *framework* Yii. La aplicación ha sido diseñada para tener una interfaz adaptable a multitud de dispositivos, haciendo especial hincapié en la versión de tabletas. Para ello se ha utilizado el *front-end responsive framework* Bootstrap.

Como resultado se ha podido mejorar el sistema utilizado previamente por la empresa. También ha ayudado en el concurso de estas playas. La aplicación acabada es capaz de gestionar todas las incidencias ocurridas en las playas, así como llevar un control del material prestado. También es capaz de llevar un seguimiento del estado meteorológico de ellas y de tener un control sobre los trabajadores.

Palabras clave

Aplicación web, meteorología, playas, socorrismo, Diseño adaptable.

Keywords

Web Application, meteorology, beaches, lifeguarding, Responsive design.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Contexto y motivación

Inttegrum Consultora y Gestora del Mediterráneo S.L., de ahora en adelante Inttegrum S.L., es una empresa dedicada al desarrollo, venta y mantenimiento de soluciones software tanto a particulares como a empresas. Además de esto, también ofrece soluciones hardware y venta de artículos de ámbito informático.

Mediterránea Grupo es uno de los clientes más importantes de la empresa. Esta empresa se dedica a la gestión de centros polifuncionales, guarderías, playas y gimnasios. Hasta ahora, la empresa ha utilizado los métodos tradicionales de gestión para sus centros. Debido a la gran cantidad de centros que debe gestionar, estos métodos tradicionales comienzan a presentar limitaciones. Cada vez es más difícil controlar todos los elementos que éstos poseen. Por ello acuden a la empresa Inttegrum S.L. para buscar una solución.

De esta idea surge el encargo de una gran aplicación dividida en varias secciones para poder gestionar todos los centros de Mediterránea Grupo. Este proyecto se centra en la gestión de las diferentes playas del litoral que la empresa tiene asignada. De estas playas se tiene que gestionar una gran cantidad de eventos, como puede ser el estado meteorológico o los partes de accidentes.

Por ello, Mediterránea Grupo decidió crear una aplicación que cubra las necesidades que sus trabajadores de las playas tienen actualmente.

1.2. Necesidades

La aplicación pretende cubrir la necesidad de mejorar su sistema de gestión actual. La gran cantidad de partes que se generan de forma diaria en las playas les obliga a mejorar su sistema de gestión de los mismos. Este nuevo sistema de gestión debe permitir llevar un seguimiento de los partes meteorológicos y de banderas por cada playa para realizar informes y observaciones de los cambios.

En estos años, el préstamo de material anfibio en las playas ha sido gestionado mediante fichas de papel. La pérdida o deterioro de estas fichas ha causado el extravío de materiales prestados. Esto ha causado problemas con los trabajadores y los encargados del préstamo de material. Por ello, la aplicación simplifica la gestión del registro del material anfibio disponible y prestado.

Otro aspecto a mejorar en cuestión de gestión es la salida de embarcaciones. Para tener constancia de la hora exacta de ida y regreso, de la tripulación y sobre todo de si está relacionada a algún incidente, esta aplicación pretende mejorar estos aspectos. Hasta ahora las fichas de papel estaban separadas, unas para incidentes y otras para salida de embarcaciones, y se tenían que contrastar entre sí para comprobar que estaban relacionadas. Los incidentes relacionados con una salida de embarcación en la aplicación deben ser accesibles desde ambas partes, siendo más sencillo comprobar información.

Todos estos requisitos deben ser accesibles mediante dispositivos táctiles móviles. En este caso, la empresa quiere que sus trabajadores utilicen tabletas en las playas para el uso de esta aplicación. Para conectarse a ella, se hará uso de tarjetas SIM[1] con conexión inalámbrica 3G UMTS[2]. Esto permitirá almacenar en tiempo real los datos de todos los partes e informes rellenados, así como poder acceder a ellos.

El cliente, aparte de necesitar la aplicación para ayudar a la gestión y automatización de tareas, también la requiere por otro motivo. Las empresas que desean gestionar las playas durante un tiempo determinado, deben presentarse a un concurso creado por los ayuntamientos a las que pertenecen esas playas. En estos concursos existe una serie de requisitos que las empresas deben cumplir para sumar puntos y poder ser elegido para la gestión. Por ello, esta aplicación pretende sumar puntos en el apartado relativo a la innovación, en el cuál se permite presentar ideas que mejoren de alguna manera la gestión.

1.3. Objetivos

Cada proyecto debe tener una serie de objetivos que deben cumplirse para que pueda darse como satisfactorio. Este proyecto, por limites temporales, es solamente una parte de un proyecto de mayor envergadura. Estos objetivos ayudan a motivar e incentivar las metas que se desean lograr con la realización de este proyecto. Por ello, los objetivos específicos de este apartado son los siguientes:

- **Gestionar un proyecto software en todas sus fases.** Esto es un objetivo importante para adquirir la experiencia necesaria en este aspecto para el futuro.
- **Aprender el uso de nuevas tecnologías.** Es importante saber adaptarse a nuevos entornos y tecnologías, por tanto esto es un objetivo a lograr.
- **Cumplir los tiempos estimados.** En todo desarrollo de software es difícil cumplir los plazos establecidos inicialmente. En este proyecto se ha propuesto como objetivo lograr cumplir la estimación inicial.

[1] Subscriber Identity Module.
[2] Universal Mobile Telecommunications System.

- **Ayudar al cliente a optimizar sus métodos de gestión.** Es importante que la aplicación cubra las necesidades e inquietudes del cliente final, a fin de lograr un proyecto satisfactorio.

Capítulo 2

Descripción del proyecto

A continuación se detallarán una serie de aspectos del proyecto. Primero, en el apartado 2.1 se explica la arquitectura general del proyecto. En el apartado 2.2 se hace un repaso a las diferentes tecnologías seleccionadas y utilizadas en el proyecto, así como la justificación de su uso.

2.1. Arquitectura del proyecto

El proyecto ha seguido una arquitectura de cliente-servidor. En esta arquitectura, la aplicación se separa entre la parte del cliente y la parte del servidor. Se encuentra dividida en tres capas diferenciadas entre sí las cuales se describirán a continuación:

- **Capa de presentación.** Cuando el usuario accede a la aplicación a través de su navegador web, interactúa directamente con esta capa. Dado que la página web es dinámica, ésta se preprocesa en el servidor antes de mostrarse al usuario.
- **Capa de lógica.** En esta capa se encuentra toda la funcionalidad a nivel de servidor de la aplicación. Esta capa interconecta las otras dos.
- **Capa de persistencia.** Aquí es donde se almacena y accede a toda la información relativa a la aplicación. La capa de lógica puede modificar, borrar, añadir y consultar esta información.

Ademas de poder diferenciarse estas tres capas, también se podrían diferenciar los niveles en la aplicación. Estos niveles representan la distribución de las capas en diferentes computadores. En este caso, dado que la capa de lógica y persistencia se encuentra en un computador, y la vista de presentación y parte de la lógica se encuentran en el cliente, se consideraría como una arquitectura a dos niveles.

De forma resumida, podríamos definir la arquitectura de esta aplicación como una **arquitectura de tres capas y dos niveles**. Se puede ver un esquema de la arquitectura en la

Figura 2.1.

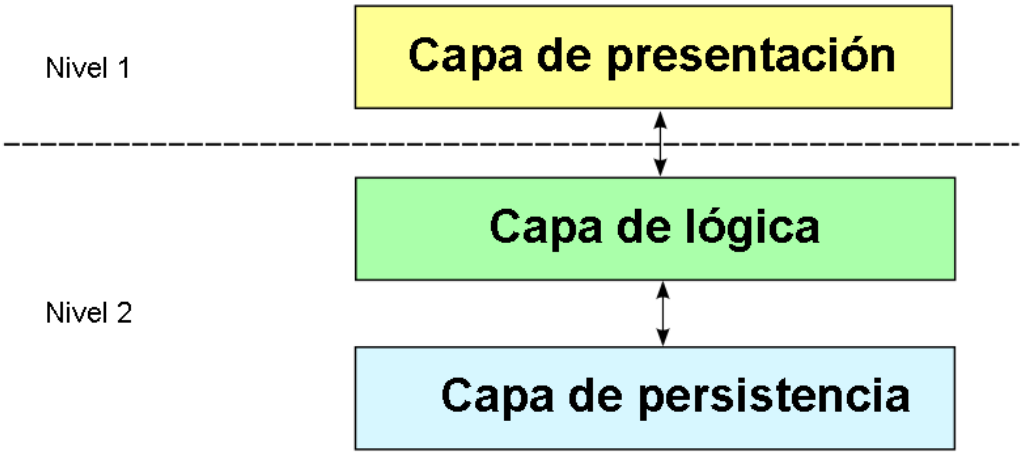

Figura 2.1: Vista general de la arquitectura de la aplicación.

2.2. Tecnologías

A lo largo del proyecto se han utilizado diversas tecnologías, tanto en el lado del cliente como en el del servidor. Es muy importante elegir bien estas tecnologías, ya que servirán como base para el funcionamiento del mismo. A continuación se detallan las utilizadas, así como la justificación de uso.

2.2.1. Servidor

Antes de desarrollar la parte de la aplicación correspondiente al lado del servidor, se tuvieron en cuenta varios requisitos previos de la empresa. Para este apartado, dado que el alojamiento web del que disponen ejecuta el lenguaje PHP [28] en un Apache HTTP server [27] y tenían documentación de un *framework* utilizado por ellos anteriormente, se optó por utilizar el mismo. Este *framework* es el Yii framework [37]. Las razones que llevaron a utilizarlo inicialmente fueron varias. Entre ellas se encuentra:

- **Seguridad.** Tiene sistemas de protección contra las vulnerabilidades web más comunes, así como un sistema de permisos y roles para los usuarios de la aplicación.
- **Generador de código personalizable.** Tiene un generador de código llamado *Gii* [36] que permite la generación del CRUD[1] de las clases directamente de la base de datos. Esto ahorra muchísimo tiempo a la hora de realizar un proyecto, especialmente si es de gestión.

[1]Create, Read, Update and Delete

- **Validación de formularios.** Este *framework* permite realizar de forma sencilla la validación de formularios, añadiendo filtros genéricos o creando propios.
- **Extensiones.** Posee una gran cantidad de extensiones y complementos para poder personalizar cualquier aspecto del *framework*.

2.2.2. Cliente

Las tecnologías utilizadas en el lado del cliente son las que se visualizan y ejecutan directamente sobre el navegador web del usuario de la aplicación. Se han utilizado las tecnologías estándares para la plataforma web: HTML [34] en su versión 5, JavaScript [16] y CSS [33] en su versión 3.

Componentes HTML, CSS y JavaScript

Para decidir qué biblioteca de componentes utilizar en el lado del cliente para la interfaz de usuario web se tuvieron en cuenta varios requisitos:

- **Interfaz adaptable.** Este criterio es un aspecto importante, dado que la aplicación se tiene que utilizar en dispositivos móviles y debe tener una interfaz que sea compatible con las distintas resoluciones y tamaños de pantalla sin perder legibilidad ni usabilidad.
- **Integrado con Yii framework**. Esto simplificaría mucho la tarea de utilizar ambas partes de forma conjunta.
- **Compatible con los navegadores más utilizados.** Es importante que la aplicación sea compatible con una gran variedad de navegadores.
- **Estética moderna.** Es importante que la interfaz de usuario se adapte a las tendencias actuales de diseño.

Tras analizar las distintas opciones que cumplieran estos aspectos, se compararon dos bibliotecas de componentes diferentes: Bootstrap [7] y Foundation [38]. Ambos son modernos, tienen diseño adaptable y están disponibles como extensiones para Yii. Las dos extensiones, tanto la de Foundation [2] como la de Yii [1] han sido creadas y mantenidas por la empresa 2amigOS [3]. Debido a la lenta actualización de las extensiones para Yii, las versiones utilizadas no son las más recientes. Para decidir cuál utilizar se realizó una comparativa mostrada en la Tabla 2.1.

Bootstrap es una biblioteca de componentes HTML, CSS y JavaScript la cual fué creada originalmente por los trabajadores de la empresa Twitter [31] para utilizarse como estándar de estilo para uso interno. Poco después fué liberado como código abierto bajo la licencia MIT [20]. Destaca por su facilidad de uso, su estilo moderno y su amplia documentación.

Foundation es la alternativa por excelencia a Bootstrap creada por la empresa Zurb. Al igual que Bootstrap, ésta también fué concebida como estándar de estilo para uso interno y fué

	Bootstrap v2.3.2	Foundation v4
Navegadores soportados	Google Chrome [15] Safari [4] Mozilla Firefox [19] Opera [21] Internet Explorer 8+ [18]	Google Chrome Safari Mozilla Firefox Internet Explorer 9+
Biblioteca JavaScript	jQuery	Zepto
Estética de elementos	Muy elaborado. Disponibles temas.	Poco elaborado. Preparado para modificarse.
Cantidad de documentación	Alta	Media

Tabla 2.1: Comparación entre Bootstrap v2.3.2 y Foundation v4.

liberado bajo licencia MIT. Destaca por su rendimiento y la facilidad a la hora de modificar su estética.

Finalmente, tras comparar ambos *frameworks* se optó por el uso de Bootstrap. Esto fué debido a varias razones:

- Se ahorra tiempo utilizando el estilo predefinido de Bootstrap, ya que en Foundation sería necesario modificarlo para que fuese más bonito.
- Tiene una gran cantidad de documentación. Esto es muy importante a la hora de comenzar el proyecto, ya que permite tener un respaldo en caso de duda.
- La extensión para Yii está más elaborada que la de su alternativa. Esto simplifica las tareas de uso en conjunto con el resto de herramientas.

Librería JavaScript

Para complementar el uso de Javascript en el cliente, se buscó una librería ligera que proporcionase complementos para simplificar las tareas necesarias. Para ello se optó por utilizar JQuery. Esta librería es de código abierto con licencia MIT, proporciona una gran cantidad de utilidades, se complementa perfectamente con el *framework* HTML seleccionado y posee una serie de *wrappers* complementarios al Yii framework. Además permite el uso de AJAX[2] para la recarga parcial de contenido.

2.2.3. Almacenamiento de datos

Para almacenar y poder acceder a la información, es necesario utilizar un sistema que sea fiable, robusto y concurrente. Por ello se decidió utilizar MySQL, un sistema gestor de bases de datos capaz de proporcionar estas características. MySQL es un sistema de código abierto liberado bajo la licencia GNU GPL [13], y posee herramientas gráficas para simplificar su uso como es el caso de PHPMyAdmin [24]. Está incluido en el paquete XAMPP utilizado para

[2]Asynchronous JavaScript And XML

pruebas, y es el único sistema gestor de bases de datos incluido en el servicio de alojamiento web utilizado para el proyecto.

2.3. Herramientas

2.3.1. Entorno de desarrollo

El entorno de desarrollo es una herramienta muy importante a la hora de realizar un proyecto software. Esta herramienta, usada correctamente, puede ahorrar muchas horas mediante la automatización de acciones repetitivas. Para seleccionar esta herramienta se requería que fuese ligera, ya que los recursos hardware disponibles eran limitados, y que proporcionase extensiones para facilitar tareas como el autocompletado o la subida del proyecto al servidor.

Por ello se optó finalmente por el Sublime Text 3 [26]. Este editor es de pago, pero dispone de una versión de prueba completamente funcional. Permite utilizar una gran cantidad de extensiones mediante su instalador de paquetes [6].

2.3.2. Gestión de tareas

Para la gestión de tareas se ha hecho uso de la herramienta Trello [29], de la empresa *Fog Creek Software* [10]. Esta herramienta es una tabla Kanban [35], en la que se pueden crear infinidad de listas. Para este proyecto se crearon tres: una de tareas pendientes (*To Do*), una de proceso de realización (*Doing*) y una de tareas terminadas (*Done*). Esta herramienta simplifica la gestión de las diferentes tareas que pueda tener el proyecto. Permite asignar autor a la tarea, poner comentarios e incluso darles prioridad. Un ejemplo del aspecto que tiene se puede ver en la Figura 2.2.

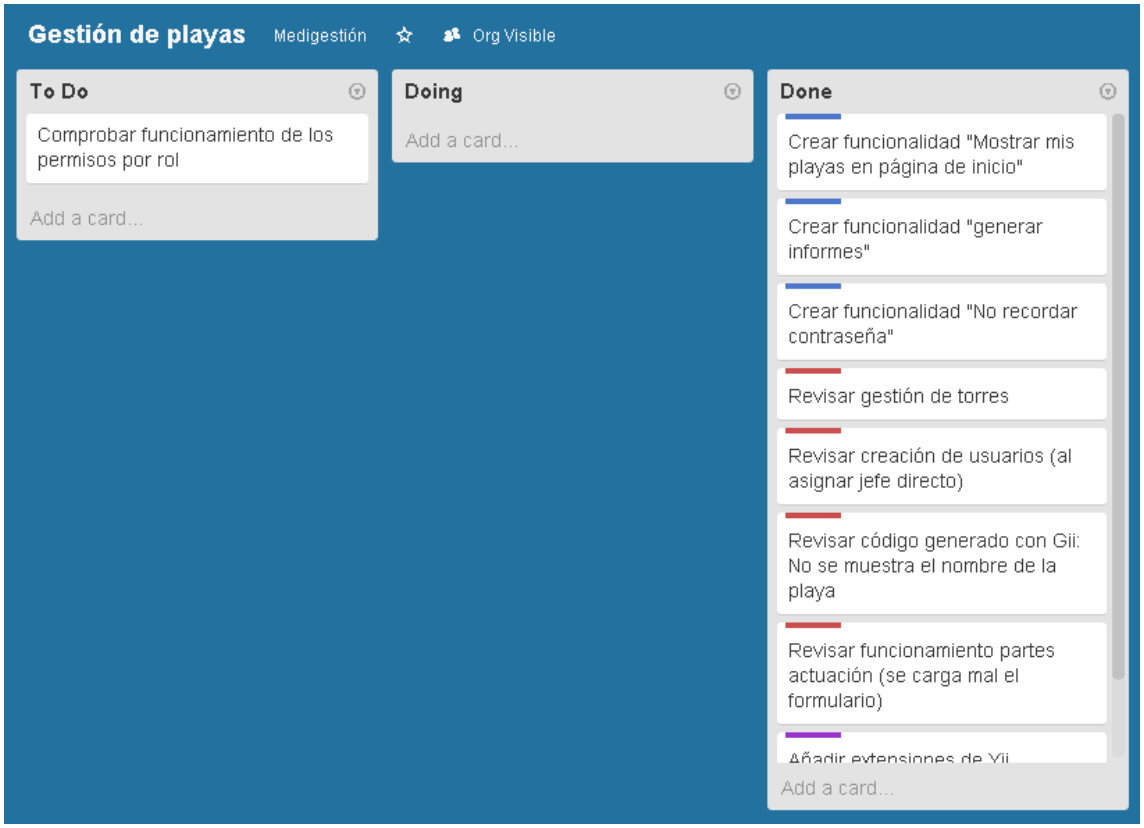

Figura 2.2: Pizarra kanban para gestión de tareas en Trello.

Capítulo 3

Planificación del proyecto

En la fase de planificación del proyecto se han asentado los requisitos y se ha comenzado a realizar estimaciones y mediciones. En el apartado 3.1 se explica la técnica de desarrollo de proyectos utilizada. En el apartado 3.2 se concretan las tareas a realizar en el proyecto. Finalmente en los apartados 3.3, 3.4 y 3.5 se realizan estimaciones de los diferentes aspectos que afectan al desarrollo de la aplicación, y que influyen como aspecto valorativo de la misma.

3.1. Metodología

La metodología de trabajo utilizada en este proyecto se basa en el modelo de desarrollo clásico tomando referencia en el *Project Management Book of Knowledge* [22], pero adaptándole algunos aspectos de la metodología ágil *KanBan*. La razón de esto es que los requisitos han sido establecidos desde el comienzo del proyecto y son invariables. No obstante, existe la posibilidad de realizar cambios menores en cuanto a preferencias estéticas o mejoras funcionales pequeñas. Esta decisión se ha tomado dada la cercanía del cliente y la posibilidad de realizar una estrecha y constante colaboración. Las fases necesarias para seguir esta metodología son:

1. **Análisis de requisitos.** En esta fase se recopilan y refinan los requisitos del sistema. Lo más importante de esta fase es conseguir que los requisitos sean claros y viables.

   En este proyecto se tomaron los requisitos existentes, se aclararon las dudas con el cliente y acordaron unos requisitos finales. Estos requisitos se pueden ver en los Casos de uso (Apartado 4.1.1) y los Requisitos no funcionales (Apartado 4.1.2).

2. **Diseño.** El objetivo de esta fase es utilizar los requisitos recopilados previamente en el análisis para adaptarlo a su forma física. Esto permite el inicio de la fase de implementación.

   En esta fase se realizó el Diagrama de clases (Apartado 5.1.2) y el Modelo conceptual de la base de datos(Apartado 5.1.1).

3. **Implementación y validación.** En la implementación se codifican los requisitos refinados en la fase de diseño. Tras esto, es importante comprobar que todo funciona correcta-

mente. Para ello, se realizan pruebas en la fase de validación.

Estas fases son necesarias para cualquier proyecto software, ya que aportan la documentación imprescindible para cualquier proyecto, ya sea en mayor o menor cantidad. Estos pasos han sido desarrollados en varias iteraciones, tomando retroalimentación constante por parte del cliente y los usuarios. A medida que ha ido avanzando el proceso de desarrollo de la aplicación, ha sido imprescindible la colaboración por parte de los mismos.

3.2. Definición de tareas

Para poder dividir el proyecto en segmentos más pequeños y estimables, se ha realizado un listado de las tareas que se requieren. En este listado se muestran todas las fases por las que el proyecto debe pasar, y las tareas estimadas de forma inicial para el mismo. Para definir con mayor exactitud el apartado del desarrollo, se ha dividido en una serie de tareas que se listarán bajo el mismo. A continuación se muestran las tareas estimadas como necesarias para este proyecto:

Identificador	**Fase**	**Tarea**
T001	Definición de requisitos	Definir requisitos de datos
T002	Definición de requisitos	Crear diagrama de casos de uso
T003	Definición de requisitos	Rellenar kanban con tareas a realizar
T004	Análisis	Crear esquema conceptual de la base de datos
T005	Análisis	Crear diagrama de clases
T006	Análisis	Validar el análisis
T007	Diseño	Crear prototipos de las interfaces de usuario
T008	Diseño	Adaptar diagrama de clases con el diseño
T009	Diseño	Retroalimentación del diseño con el cliente
T010	Implementación y validación	Programación de las funcionalidades esenciales
T011	Implementación y validación	Programación de módulos adicionales
T012	Implementación y validación	Pruebas unitarias
T013	Implementación y validación	Pruebas de aceptación del cliente
T014	Implementación y validación	Implantar en el sistema real
T015	Implementación y validación	Entrega y formación del cliente

Tabla 3.1: Tareas del proyecto.

3.3. Estimación temporal

Para la estimación de la duración del proyecto y sus diferentes fases, se ha realizado un diagrama de Gantt. En la Figura 3.1 se puede ver la estimación inicial realizada teniendo en cuenta todos los aspectos del proyecto. Como se disponía de una cantidad de horas superior a las de la asignatura relacionada con este proyecto debido a una beca concedida, se optó por invertir una mayor cantidad en el desarrollo.

	EDT	Nombre	Esfuerzo	Predecesoras
1	**1**	**⊟Desarrollo de la propuesta técnica**	**52h**	
2	**1.1**	**⊟Inicio**	**8h**	
3	1.1.1	Definir el proyecto con el tutor y el supervisor	1h	
4	1.1.2	Definir la metodología de trabajo	5h	3
5	1.1.3	Definir estándares de trabajo	2h	3
6	**1.2**	**⊟Documentación y planificación del proyecto**	**30h**	**2**
7	1.2.1	Documentarse sobre el contexto	12h	
8	1.2.2	Documentarse sobre los requisitos	12h	7
9	1.2.3	Identificar los objetivos	5h	8
10	1.2.4	Identificar el alcance	1h	8
11	**1.3**	**⊟Planificación del proyecto**	**14h**	**6**
12	1.3.1	Definir tareas y estimar fechas	6h	
13	1.3.2	Crear diagrama de Gantt	2h	12
14	1.3.3	Redactar la propuesta del proyecto	6h	13
15	1.3.4	Entregar la propuesta técnica	0h	14
16	**2**	**⊟Desarrollo del proyecto**	**364h**	**1**
17	**2.1**	**⊟Definición de requisitos**	**23h**	
18	2.1.1	Definir requisitos de datos	12h	
19	2.1.2	Crear diagrama de casos de uso	6h	
20	2.1.3	Rellenar kanban con las tareas a realizar	5h	18,19
21	**2.2**	**⊟Análisis**	**42h**	**17**
22	2.2.1	Crear esquema conceptual de la base de datos	20h	
23	2.2.2	Crear diagrama de clases	18h	
24	2.2.3	Validar el análisis	4h	22,23
25	**2.3**	**⊟Diseño**	**23h**	**21**
26	2.3.1	Crear prototipos de las interfaces de usuario	12h	
27	2.3.2	Adaptar diagrama de clases con el diseño	8h	
28	2.3.3	Retroalimentación del diseño con el cliente	3h	
29	**2.4**	**⊟Implementación**	**220h**	**25**
30	2.4.1	Programación de las funcionalidades esenciales	180h	
31	2.4.2	Programación de módulos adicionales	40h	30
32	**2.5**	**⊟Pruebas**	**28h**	
33	2.5.1	Pruebas unitarias	20h	
34	2.5.2	Pruebas de aceptación del cliente	8h	
35	**2.6**	**⊟Implantación**	**28h**	**29,32**
36	2.6.1	Implantar en el sistema real	20h	
37	2.6.2	Entrega y formación del cliente	8h	36
38	**3**	**⊟Documentación y presentación del TFG**	**138h**	**1**
39	3.1	Redacción de informes quincenales	8h	
40	3.2	Redacción y entrega de la memoria final	110h	
41	3.3	Preparación de la presentación oral	20h	40
42	3.4	Presentación oral	0h	41

Figura 3.1: Diagrama Gantt de estimación temporal.

3.4. Estimación de recursos

A la hora de realizar este proyecto, han sido necesarios una serie de recursos físicos. Estos incluyen los recursos hardware y software involucrados tanto para su realización como para su despliegue. A continuación se detallan las especificaciones técnicas de estos recursos.

3.4.1. Recursos hardware

Los recursos hardware son los recursos relacionados con dispositivos físicos. En este caso, se ha incluido la contratación del alojamiento web como parte de recursos hardware dado que dispone de un servidor web.

Computador de sobremesa. Este recurso ha sido utilizado para la implementación del proyecto por el alumno y como servidor local de pruebas. Sus especificaciones técnicas son:

- CPU Intel Pentium 4 HT SL6WK 3.0 GHz
- Memoria RAM 1 GB DDR2
- Tarjeta gráfica ATI Radeon 9550 256 MD DDR
- Disco duro 100 GB Seagate
- Sistema operativo Windows

Servidor. Este servidor pertenece al servicio de alojamiento web contratado. Los detalles del tipo de alojamiento son:

- 100 GB de espacio web.
- Memoria RAM hasta 0.2 GB.
- Sistema operativo Windows

Tableta bq Edison 2. En este dispositivo se han probado los diseños adaptables. Sus especificaciones son:

- CPU Quad Core Cortex A9 1.6GHz
- Memoria RAM 2 GB
- Almacenamiento 32 GB ampliable con tarjeta SD [1]
- Pantalla 10.1 pulgadas
- Sistema operativo Android 4.2.2

[1] Secure Digital

3.4.2. Recursos software

Estos recursos engloban todos los programas y sistemas operativos utilizados. A continuación se describen los utilizados:

Microsoft Windows 7. Sistema operativo de licencia privativa utilizado tanto en el servidor como en el computador utilizado para la implementación.

Android 4.2.2. Sistema operativo libre con licencia Apache Software License v2 [12] de dispositivos móviles utilizado en las tabletas.

Sublime Text 3. Editor de texto utilizado para la implementación del proyecto. Utilizado con licencia gratuita.

Inkscape. Herramienta gratuita de edición de imágenes vectoriales utilizado para el diseño de algunos diagramas.

MySQL Workbench. Herramienta gratuita de gestión de bases de datos que permite la edición de diagramas entidad-relación.

PHPMyAdmin. Herramienta gratuita para gestionar bases de datos desde una interfaz web. Funciona con una instancia de PHP y permite editar de forma visual una base de datos MySQL en tiempo real.

Gantter. Herramienta online para crear diagramas de Gantt. Permite guardar los proyectos realizados y exportarlos a otro tipo de herramientas, como Microsoft Project.

Draw.io. Extensión de Google Docs que permite la edición y almacenamiento de diagramas de todo tipo. Ha sido utilizado para crear los diagramas de casos de uso y de clases.

Trello. Pizarra KanBan ideal para la gestión de proyectos. Permite crear y guardar tablas personalizadas.

Balsamiq. Esta herramienta permite realizar *mockups* de una forma sencilla y directa. Ha sido utilizada para la realización de todos los prototipos de la aplicación.

Biblioteca PHPMail. Biblioteca que permite el envío de correos a través de PHP. Es gratuita y ha sido utilizada como complemento para el *framework* Yii.

Biblioteca PHPExcel. Librería que permite generar hojas de cálculo a través de PHP. Es gratuita y ha sido utilizada como complemento para el *framework* Yii.

3.5. Estimación de costes

Los costes de un proyecto son un aspecto muy importante a tener en cuenta. Este factor puede influir en si es viable y si se debe seguir adelante con él. Para estimar el coste del proyecto, se ha hecho suposición de un sueldo medio de programador junior en España en 2014 [30] para el alumno realizador del proyecto. Este coste no es real, ya que la estancia en prácticas no ha sido remunerada por parte de la empresa. A continuación se muestra una estimación de los costes:

- **Licencia Windows 7 SP1 64 Bits** → 110 €
- **Computador de sobremesa** → 500 €
- **Servidor contratado de alojamiento web** → Contratación anual: 4,99 € x 12 meses
- **Tableta bq Edison 2** → 180 €
- **Alumno en prácticas** → 7.5 € brutos x 364 horas = 2730 €

De estos valores calcularemos la contingencia de la Seguridad Social en 2014 para el sueldo [25], siendo esta 28,3 %. Como resultado obtenemos 3502,59 €.

Tras estimar los gastos necesarios para realizar el proyecto en cuestión, se puede decir que el coste final ascenderá a la cantidad de **4.352,47 €**. Este precio no es el final, pues se debe prorratear el coste del material especificado y llegar a un acuerdo de cuánto asume el cliente, alrededor de un 15 %. Debe recalcarse que es un precio estimado y que no incluye gastos adicionales dependientes de riesgos como podría ser una avería del computador de sobremesa o de la tableta.

Capítulo 4

Análisis

En esta fase se recopilan los requisitos, y se refinan y clarifican para que resulten óptimos para su utilización para el proyecto. De esta fase se obtienen los casos de uso (Apartado 4.1.1), que son un resultado de este refinamiento de requisitos, así como una serie de requisitos no funcionales (Apartado 4.1.2) que complementan a los casos de uso.

4.1. Definición de requisitos

A la hora de definir los requisitos de la aplicación, es importante tener claros cuales son los objetivos que se quieren conseguir con esta aplicación y qué necesidades se pretenden cubrir. Esto ayuda a clarificar las ideas que el cliente pueda tener inicialmente, para guiarle hacia el producto que realmente cubra lo que él necesita sin exceder el presupuesto que se tenga. En este proyecto se tuvo un contacto cercano con el cliente, por tanto se pudo clarificar estas necesidades de forma ágil y sencilla.

4.1.1. Casos de uso

Los casos de uso se definen como las acciones o procesos que el sistema debe poder realizar, formando parte de los requisitos funcionales de un sistema. Estos casos de uso están relacionados a los actores, que son los agentes que pueden interactuar con el sistema. Para esta aplicación se han definido una serie de casos de uso relacionándolos con los actores definidos previamente. Estos actores, disponibles en la tabla 4.1, son los roles que la aplicación permite utilizar de forma interna. Un usuario de la aplicación puede estar ligado a más de un rol.

[1]Diplomado Universitario en Enfermería

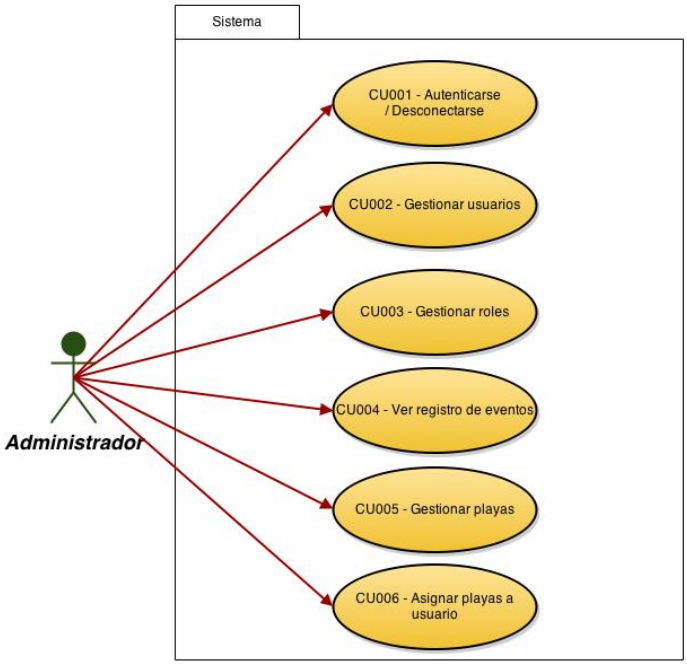

Figura 4.1: Casos de uso. Administrador.

Actor	**Descripción**
Administrador	Es el encargado de gestionar la aplicación por completo. Posee permisos para realizar cualquier acción.
Baño asistido	Es un trabajador de las playas que únicamente tiene permisos para el préstamo de material.
DUE [1]	Es un trabajador de las playas diplomado en enfermería. Se encarga de realizar curas en las playas y posee permisos para los apartados relacionados con esto.
Jefe de playa	Es el encargado de una playa. Posee permisos para asignar roles a sus trabajadores y gestionar todo lo relacionado con la playa o playas a las que esté asignado.
Patrón	Es un trabajador de las playas dedicado al uso de las embarcaciones. Tiene permisos para las acciones relacionadas con embarcaciones.
Socorrista	Es un trabajador de las playas dedicado al socorrismo. Tiene permisos para una variedad de acciones, dado que su rol tiene una variedad de obligaciones.

Tabla 4.1: Actores.

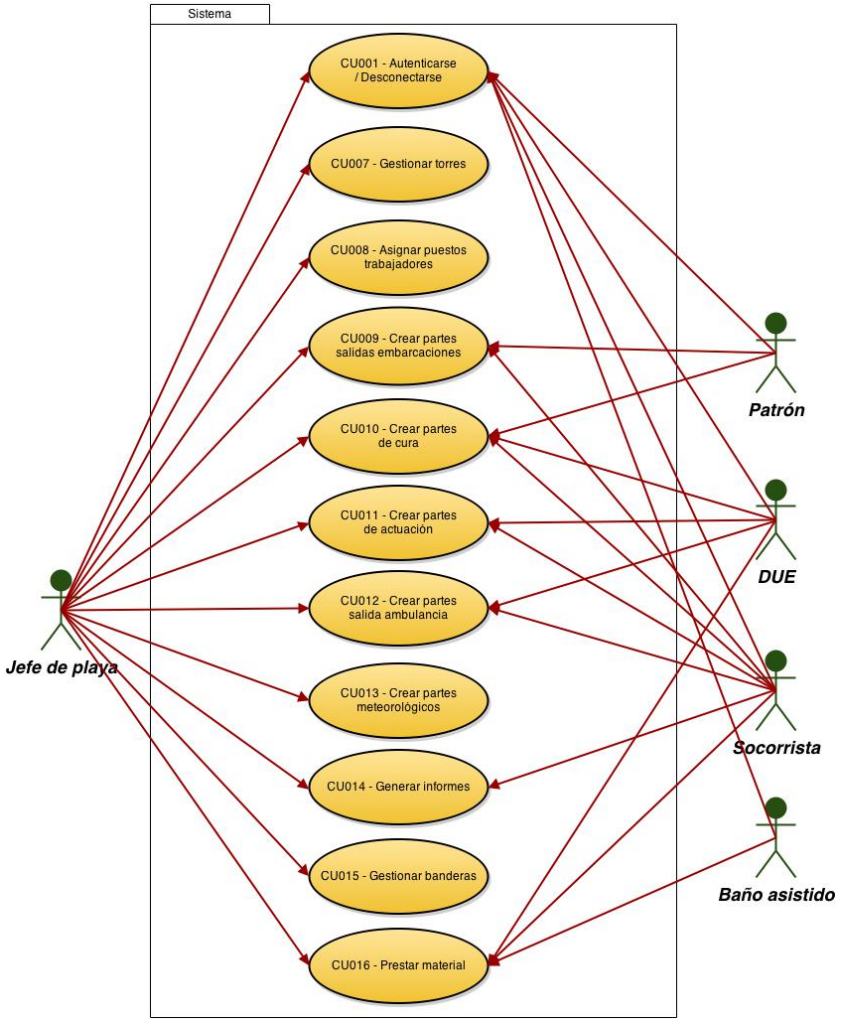

Figura 4.2: Casos de uso. Usuarios.

Identificador	CU001
Caso de uso	Autenticarse / Desconectarse
Actores	Administrador, Baño asistido, DUE, Jefe de playa, Patrón, Socorrista
Casos relacionados	Ninguno
Descripción	El sistema debe permitir a los usuarios acceder a la aplicación usando sus credenciales.
Precondición	El usuario debe estar registrado en el sistema.
Prioridad	Alta

Tabla 4.2: Caso de uso CU001.

Identificador	CU002
Caso de uso	Gestionar usuarios
Actores	Administrador
Casos relacionados	Ninguno
Descripción	El sistema debe permitir dar de alta, baja, consultar o modificar a un usuario.
Precondición	En caso de alta, no debe existir. En baja, consulta o modificación debe existir previamente.
Prioridad	Alta

Tabla 4.3: Caso de uso CU002.

Identificador	CU003
Caso de uso	Gestionar roles
Actores	Administrador
Casos relacionados	CU002
Descripción	El sistema debe permitir asignar roles a un usuario.
Precondición	El usuario debe existir previamente.
Prioridad	Alta

Tabla 4.4: Caso de uso CU003.

Identificador	CU004
Caso de uso	Ver registro de eventos
Actores	Administrador
Casos relacionados	Todos
Descripción	El sistema debe permitir ver un registro de los eventos que van ocurriendo. Estos eventos pueden ser errores o acciones, y sirve para tener un control de lo que ocurre en la aplicación.
Precondición	Ninguna
Prioridad	Baja

Tabla 4.5: Caso de uso CU004.

Identificador	CU005
Caso de uso	Gestionar playas
Actores	Administrador
Casos relacionados	Ninguno
Descripción	El sistema debe permitir añadir nuevas playas. También debe permitir borrar o modificar las playas disponibles.
Precondición	Para modificar o borrar una playa, debe existir previamente. Para añadir una nueva, no debe existir.
Prioridad	Alta

Tabla 4.6: Caso de uso CU005.

Identificador	CU006
Caso de uso	Asignar playas a usuario
Actores	Administrador
Casos relacionados	CU005
Descripción	El sistema debe permitir asignar una o múltiples playas a los usuarios. Esto les permite acceder a las acciones posibles para su rol y playa.
Precondición	Tanto el usuario como la playa deben estar registrados previamente.
Prioridad	Alta

Tabla 4.7: Caso de uso CU006.

Identificador	CU007
Caso de uso	Gestionar torres
Actores	Administrador, Jefe de playa
Casos relacionados	CU005
Descripción	El sistema debe permitir asignar y borrar torres a una playa. Las torres representan donde está situado un socorrista.
Precondición	La playa debe estar registrados previamente.
Prioridad	Media

Tabla 4.8: Caso de uso CU007.

Identificador	CU008
Caso de uso	Asignar puestos a trabajadores
Actores	Administrador, Jefe de playa
Casos relacionados	CU005
Descripción	El sistema debe permitir asignar roles a usuarios en una playa específica. Un usuario puede tener múltiples roles.
Precondición	La playa debe estar registrada previamente. Tanto el jefe de playa como el usuario objetivo deben estar asignados a la playa.
Prioridad	Media

Tabla 4.9: Caso de uso CU008.

Identificador	CU009
Caso de uso	Crear partes salidas embarcaciones
Actores	Administrador, Jefe de playa, Patrón, Socorrista
Casos relacionados	CU011
Descripción	El sistema debe permitir registrar las salidas de las embarcaciones. También se podrán ver las salidas realizadas, así como su asociación con alguna actuación o cura.
Precondición	La playa debe estar registrada previamente. Tanto el jefe de playa como el usuario objetivo deben estar asignados a la playa.
Prioridad	Media

Tabla 4.10: Caso de uso CU009.

Identificador	CU010
Caso de uso	Crear partes de cura
Actores	Administrador, Jefe de playa, DUE, Patrón, Socorrista
Casos relacionados	CU012
Descripción	El sistema debe permitir registrar las curas realizadas en las playas.
Precondición	La playa debe estar registrada previamente. El usuario debe estar registrado previamente.
Prioridad	Media

Tabla 4.11: Caso de uso CU010.

Identificador	CU011
Caso de uso	Crear partes de actuación
Actores	Administrador, Jefe de playa, DUE, Socorrista
Casos relacionados	CU009, CU012
Descripción	El sistema debe permitir registrar las actuaciones realizadas en las playas. Puede estar relacionado o no con los partes de salidas de embarcaciones y de ambulancias, dependiendo de la actuación en cuestión.
Precondición	La playa debe estar registrada previamente. El usuario debe estar registrado previamente.
Prioridad	Media

Tabla 4.12: Caso de uso CU011.

Identificador	CU012
Caso de uso	Crear partes salida ambulancia
Actores	Administrador, Jefe de playa, DUE, Socorrista
Casos relacionados	CU011, CU010
Descripción	El sistema debe permitir registrar las salidas de ambulancia que se produzcan.
Precondición	La playa debe estar registrada previamente. El usuario debe estar registrado previamente.
Prioridad	Media

Tabla 4.13: Caso de uso CU012.

Identificador	CU013
Caso de uso	Crear partes meteorológicos
Actores	Administrador, Jefe de playa
Casos relacionados	CU005
Descripción	El sistema debe permitir registrar las condiciones meteorológicas de la playa.
Precondición	La playa debe estar registrada previamente. El usuario debe estar registrado previamente.
Prioridad	Media

Tabla 4.14: Caso de uso CU013.

Identificador	CU014
Caso de uso	Generar informes
Actores	Administrador, Jefe de playa, Socorrista
Casos relacionados	Todos
Descripción	El sistema debe permitir generar informes de todos los aspectos relacionados con las playas y sus partes. Esto permite realizar un seguimiento constante a la situación de las mismas.
Precondición	La playa debe estar registrada previamente. El usuario debe estar registrado previamente. Deben existir registros de los eventos en cuestión.
Prioridad	Baja

Tabla 4.15: Caso de uso CU014.

Identificador	CU015
Caso de uso	Gestionar banderas
Actores	Administrador, Jefe de playa
Casos relacionados	CU005, CU007
Descripción	El sistema debe permitir asignar banderas a las torres de las playas. También debe permitir ver un histórico de los cambios de banderas.
Precondición	La playa debe estar registrada previamente. El usuario debe estar registrado previamente. Debe tener la playa asignada.
Prioridad	Media

Tabla 4.16: Caso de uso CU015.

Identificador	CU016
Caso de uso	Prestar material
Actores	Administrador, Jefe de playa, Socorrista, Baño asistido, DUE
Casos relacionados	CU005
Descripción	El sistema debe permitir registrar el préstamo de material anfibio de las playas.
Precondición	La playa debe estar registrada previamente. El usuario debe estar registrado previamente. Debe tener la playa asignada.
Prioridad	Media

Tabla 4.17: Caso de uso CU016.

4.1.2. Requisitos no funcionales

A la hora de especificar los requisitos, existen una serie de ellos que no forman parte de las acciones y funcionalidades directas de la aplicación. Por ello, este apartado recoge los requisitos que son imprescindibles para la aplicación, pero que no corresponden a casos de uso.

Identificador	RNF001
Caso de uso	Diseño adaptable
Descripción	La interfaz gráfica del sistema debe ser adaptable a dispositivos móviles y a diferentes resoluciones de pantalla.

Tabla 4.18: Requisito no funcional RNF001.

Identificador	RNF002
Caso de uso	Aplicación ligera
Descripción	El sistema debe ser ligero en cuanto a tiempo de bajada, ya que se pretende utilizar desde conexiones de datos móviles.

Tabla 4.19: Requisito no funcional RNF002.

Identificador	RNF003
Caso de uso	Estabilidad del sistema
Descripción	El sistema debe ser estable para evitar la repetición de las tareas.

Tabla 4.20: Requisito no funcional RNF003.

Identificador	RNF004
Caso de uso	Usabilidad de la aplicación
Descripción	El sistema debe estar diseñado para ser fácilmente usable por todo tipo de usuarios.

Tabla 4.21: Requisito no funcional RNF004.

Capítulo 5

Diseño

A la hora de realizar la fase de diseño, se recogen los requisitos refinados en la fase de análisis y se convierten en elementos más cercanos a la fase de implementación. Los datos necesarios para la aplicación se modelan y plasman en el modelo conceptual de datos (Apartado 5.1.1), mientras que las clases y objetos necesarios se muestran en el diagrama de clases (Apartado 5.1.2).

5.1. Diseño del sistema

El diseño del sistema comprende el modelado de los elementos necesarios para el funcionamiento interno de la aplicación, su lógica y su persistencia. De ello se puede extraer los esquemas y conceptos mostrados a continuación.

5.1.1. Modelo conceptual de datos

A la hora de diseñar la base de datos, se han tenido que cumplir ciertas normas impuestas por el Yii framework. La mas importante es que todas las tablas deben tener un campo llamado ID y que éste sea la clave primaria. A pesar de ser a veces innecesario, como es el caso de una relación muchos a muchos, el *framework* en cuestión obliga a hacer uso de esta clave. Esto es debido a que utiliza una abstracción interna de la base de datos para simplificar su acceso y uso. Este componente, el *CActiveRecord*, mapea todos los modelos de la base de datos conectada y los enlaza entre sí. Este tipo de mapeo es conocido también como un ORM [1] y simplifica mucho el uso de la base de datos. Por ello, se optó por seguir los estándares impuestos, a pesar de no corresponder totalmente con un modelado de la base de datos idílico.

Para el diseño del diagrama se ha hecho uso de la herramienta MySQL Workbench. Esta herramienta es muy versátil y permite editar de forma gráfica el modelado de una base de datos MySQL. También permite la exportación e importación de la misma, por tanto ha sido posible extraer el SQL correspondiente al modelado. Se ha seguido la notación *Crow's Foot* [32] para

[1] Object-Relational Mapping

el diseño del diagrama de las Figuras 5.1 y 5.2.

Como estándar de notación para las tablas de las playas, se ha tomado como primera palabra la sección a la que pertenece seguido de la tabla con barras bajas de separación. Al ser muy largos los nombres, se ha simplificado tomando las tres primeras letras de cada palabra. Para la parte de los usuarios y centros no ha sido necesario, ya que los nombres eran mucho mas cortos. La base de datos está dividida en dos partes: usuarios y administrador. A pesar de ello en la implementación no se ha dividido con el fin de simplificar su visualización y comprensión. En las tablas 5.1 y 5.2 se pueden ver los nombres completos, así como una breve descripción de su función para la aplicación.

Muchas de las tablas poseen un atributo *autor_id*. Este atributo sirve para guardar un registro del autor del evento en la base de datos, independientemente de para quién se haya realizado.

Tabla	**Nombre completo**	**Descripción**
ply_par_sal_emb	playa parte salida embarcación	Tabla para registrar las salidas de embarcaciones.
ply_par_sal_emb_usu_tra	playa parte salida embarcación usuario trabajo	Tabla para ver que usuarios han salido en las salidas de embarcaciones.
ply_par_cur	playa parte cura	Tabla para registrar los partes de curas.
ply_par_act	playa parte actuación	Tabla para registrar los partes de actuación.
ply_par_sal_amb	playa parte salida ambulancia	Tabla para registrar las salidas de las ambulancias.
ply_par_uso_mat_anf	playa parte uso material anfibio	Tabla para registrar la utilización de material anfibio.
ply_reg_met	playa registro meteorológico	Tabla para llevar el registro meteorológico.
ply_reg_met_vie	playa registro meteorológico viento	Tabla para llevar el registro meteorológico del viento.
ply_par_ban	playa parte bandera	Tabla para registrar las banderas.
ply_tor	playa torre	Tabla para registrar las distintas torres de una playa.

Tabla 5.1: Tablas de la base de datos de playas.

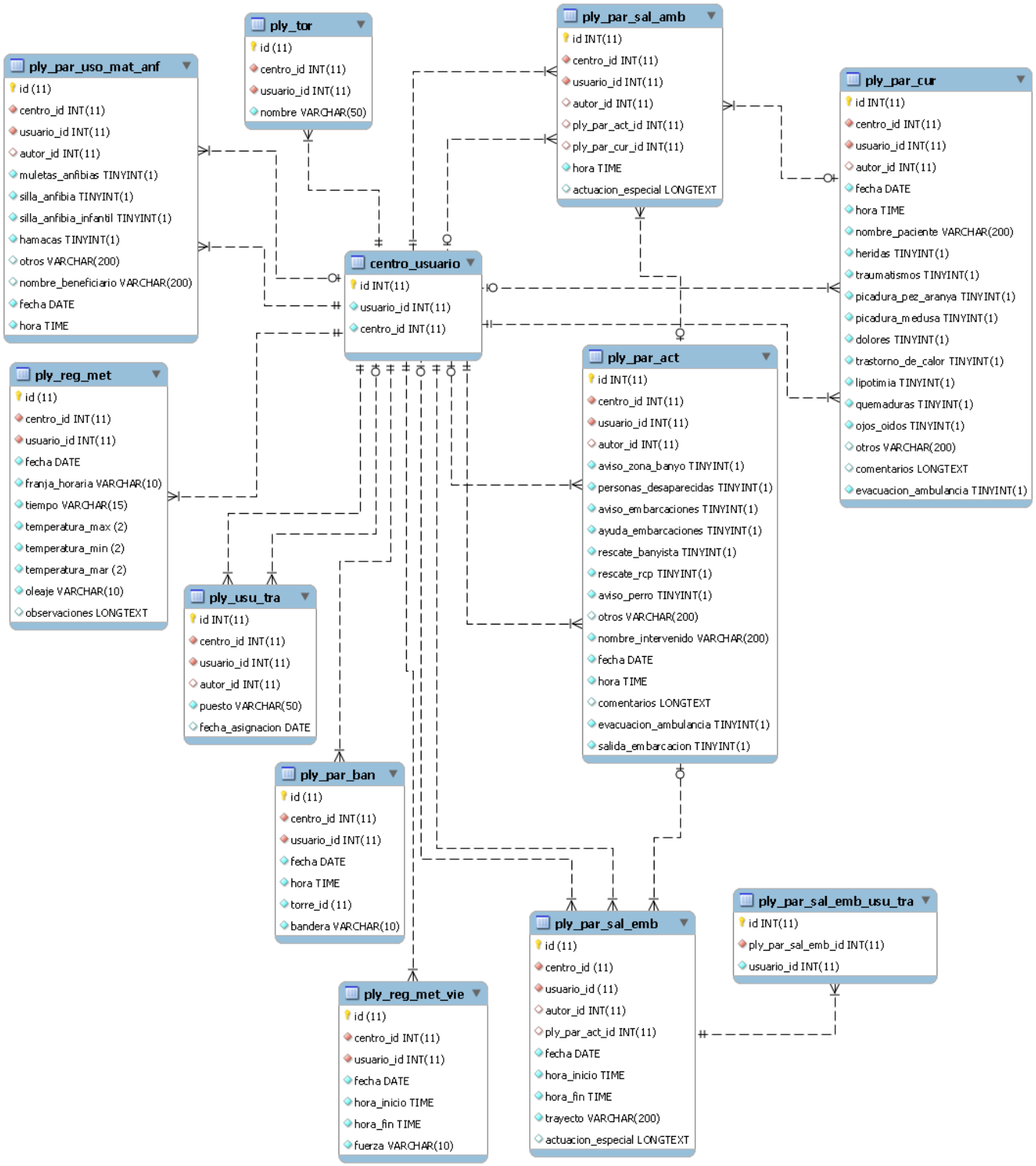

Figura 5.1: Modelo conceptual de la base de datos de playas.

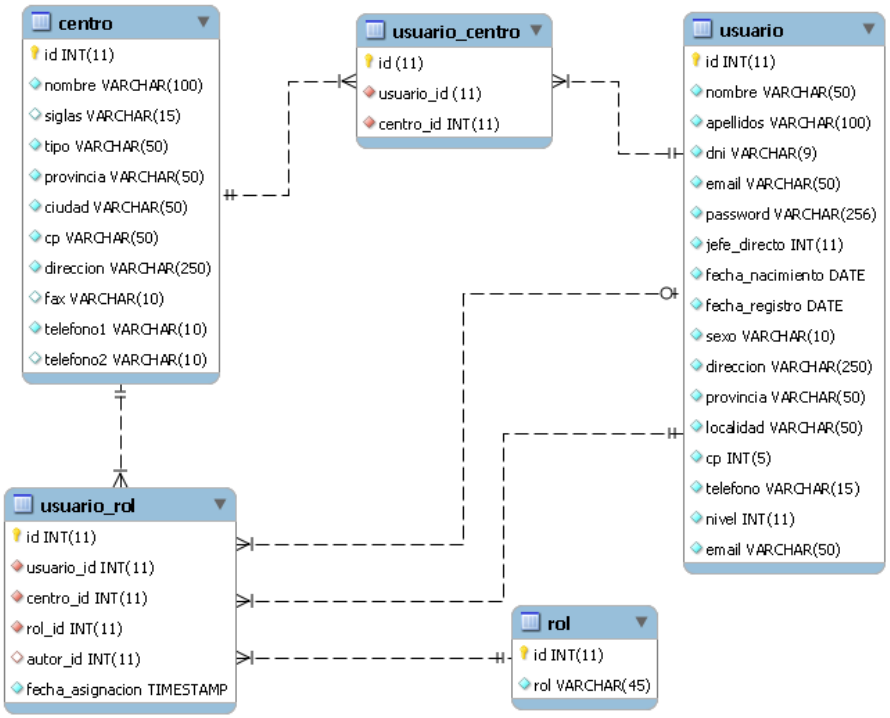

Figura 5.2: Modelo conceptual de la base de datos de usuarios y centros.

Tabla	**Descripción**
usuario_centro	Tabla para relacionar un usuario a una playa.
usuario	Tabla para almacenar los usuarios de la aplicación.
centro	Tabla para almacenar las playas disponibles. Se llama centro para almacenar en un futuro diferentes tipos de centros.
usuario_rol	Tabla que relaciona un usuario con un rol. Esto sirve para definir los permisos.
rol	Tabla para almacenar los diferentes roles que puede tener un usuario.

Tabla 5.2: Tablas de la base de datos de usuarios y centros.

5.1.2. Diagrama de clases

El diagrama de clases de la aplicación, al utilizarse el Yii framework, corresponde directamente con el modelo conceptual de la base de datos presentado anteriormente. Habitualmente existen algunas diferencias entre ambos diagramas, al no realizarse una conversión directa de la base de datos a las clases del lenguaje seleccionado. No obstante, en este caso el framework mapea todas las tablas de la base de datos y utiliza un generador para crear una clase correspondiente a cada una de ellas. Por ello, se ha optado por mostrar el diagrama de clases de una entidad en particular (Figura 5.3), siendo esta extrapolable al resto de entidades de la base de datos.

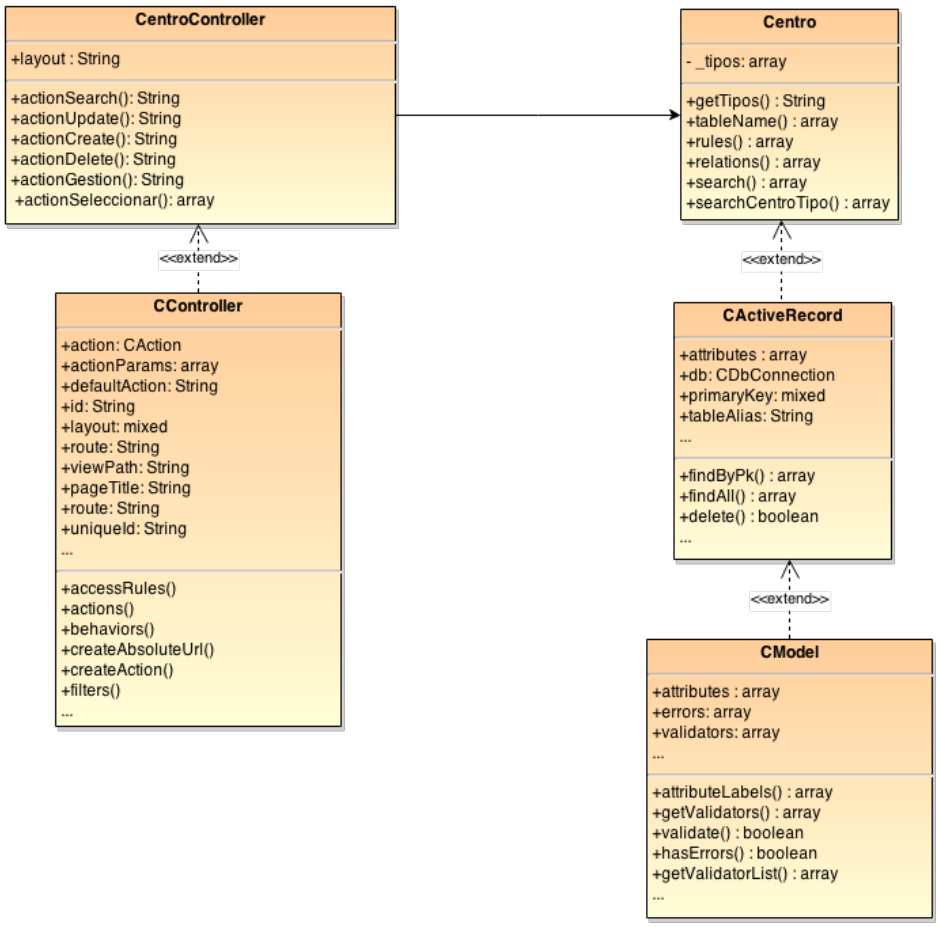

Figura 5.3: Diagrama de clases. Clase centro.

5.2. Diseño de la interfaz

A la hora de elegir qué diseño utilizar, era imprescindible el uso de un diseño adaptable. Para ello, como se explica anteriormente en el apartado 2.2.2, se optó por utilizar Bootstrap. Esta biblioteca permite utilizar *layouts* adaptables a dispositivos móviles sin complicación alguna. Por tanto, a la hora de diseñar prototipos, se tuvo muy en cuenta la biblioteca de componentes a utilizar y se intentó crear un diseño que fuese simple, intuitivo y adaptable.

Para diseñar la interfaz, se siguieron cuatro pautas básicas [5]. Estas pautas se describen a continuación:

- **Proximidad.** Los elementos que se agrupan entre sí dan la sensación de pertenecer a un único ente o grupo. Este principio se basa en que el cerebro humano asocia las agrupaciones de elementos a esto, y permite separar categorías de elementos entre sí. Esto se puede apreciar en la agrupación de los campos de los formularios o elementos de los menús.
- **Alineación.** Otro aspecto importante es la alineación de los elementos entre sí. Si los elementos se posicionan sin tener relación entre sí, da una sensación de caos en la interfaz. Como la biblioteca Bootstrap utiliza un sistema de distribución de espacio llamado *940 grid system*. Este sistema consiste en dividir 940 píxeles en 12 columnas, e ir encajando en esas columnas los elementos. De esta forma, todos los elementos están posicionados de una forma homogénea y coherente.
- **Repetición.** Es importante repetir de forma consistente los tipos de fuentes, los colores y los estilos de la aplicación en general. Si se varía mucho, puede parecer desordenado. En esta aplicación se han utilizado estilos homogéneos a lo largo de la aplicación.
- **Contraste.** Es importante tener contraste entre los elementos del diseño. Elementos que sean similares pueden dar lugar a confusión. En la aplicación, los elementos como los botones contrastan mucho con el fondo para dar esa diferenciación clave.

Además de seguir estos principios, se optó por utilizar una biblioteca de imágenes SVG [2]. Esto permite utilizar imágenes escalables a cualquier resolución, ideal para una aplicación como ésta que requiere un diseño responsivo. La biblioteca se llama *FontAwesome* [14], está disponible utilizando una combinación de las licencias MIT, CC 3.0 [9] y SIL open font [17], permitiéndole ser totalmente gratuita para uso comercial. Destaca por su facilidad de uso, ya que se utiliza como una fuente de texto cualquiera y se puede modificar mediante CSS como si fuese una.

5.2.1. Prototipos

Los prototipos de la aplicación son una primera versión de las interfaces gráficas que van a estar disponibles en la aplicación. Tomándolas como base, se realizaron pequeñas modificaciones en la versión final, pero siempre manteniendo la esencia del prototipo desarrollado. Estos prototipos están mostrados por orden, comenzando por el acceso (Figura 5.4). Esta interfaz esta pensada para ser simple y directa, sin distracciones de por medio.

[2] Scalable Vector Graphics

Una vez se entra en la aplicación, se selecciona la playa a la que se quiere acceder (Figura 5.5). Esto es debido a que un usuario puede ser trabajador en múltiples playas, y cada día puede variar la que necesita. Se optó por mostrar un listado con todas las asociadas al usuario junto con un botón que permita acceder a ella.

El menú principal de la aplicación está compuesto por una serie de iconos con descripción, los cuales representan las partes esenciales de la aplicación (Figura 5.6). Está diseñada de forma sencilla de utilizar desde tablets, adaptándose los iconos al tamaño estándar de pantalla.

Tras esto se diseñó uno de los apartados de gestión de la aplicación, ya que los demás compartirían el mismo aspecto (Figura 5.7). Se optó por poner un listado de las playas disponibles para no tener que salir y entrar de la aplicación en cada momento para cambiar de playa. Esto es debido a que los usuarios de la aplicación explicaron que muchas veces tenían que cambiar entre playas para ir realizando diferentes partes o consultas.

Finalmente se diseñó un *widget* desde cero, pensado para simplificar la creación de informes de banderas (Figura 5.8). Este *widget* se compone de un seleccionador de banderas para elegir el tipo de bandera que tiene cada torre de una playa en un momento determinado. Aunque en el prototipo no se pueda apreciar, cada icono de bandera estaba pensado para ser del color que representa. De esta manera simplifica mucho la tarea de poner banderas a las diferentes playas de forma diaria.

Para ver el resultado final tras haber implementado estos prototipos en el proyecto, se puede consultar el anexo A.

Figura 5.4: Prototipo. Acceso.

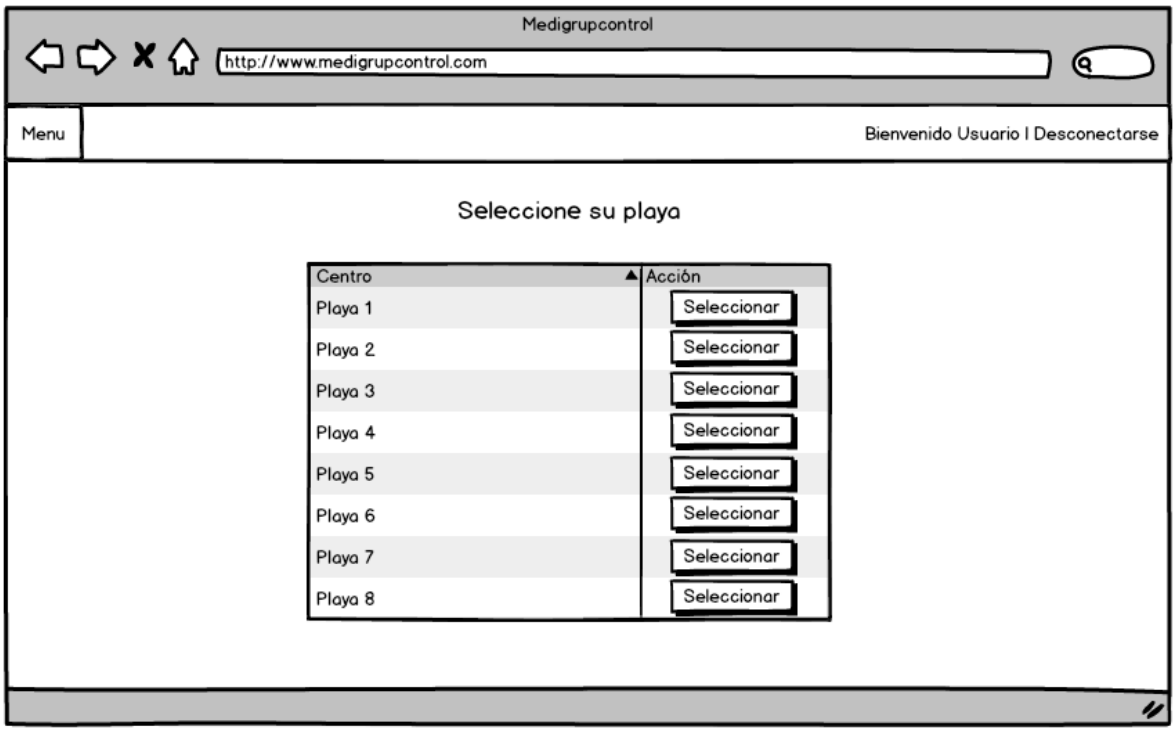

Figura 5.5: Prototipo. Seleccionar centro.

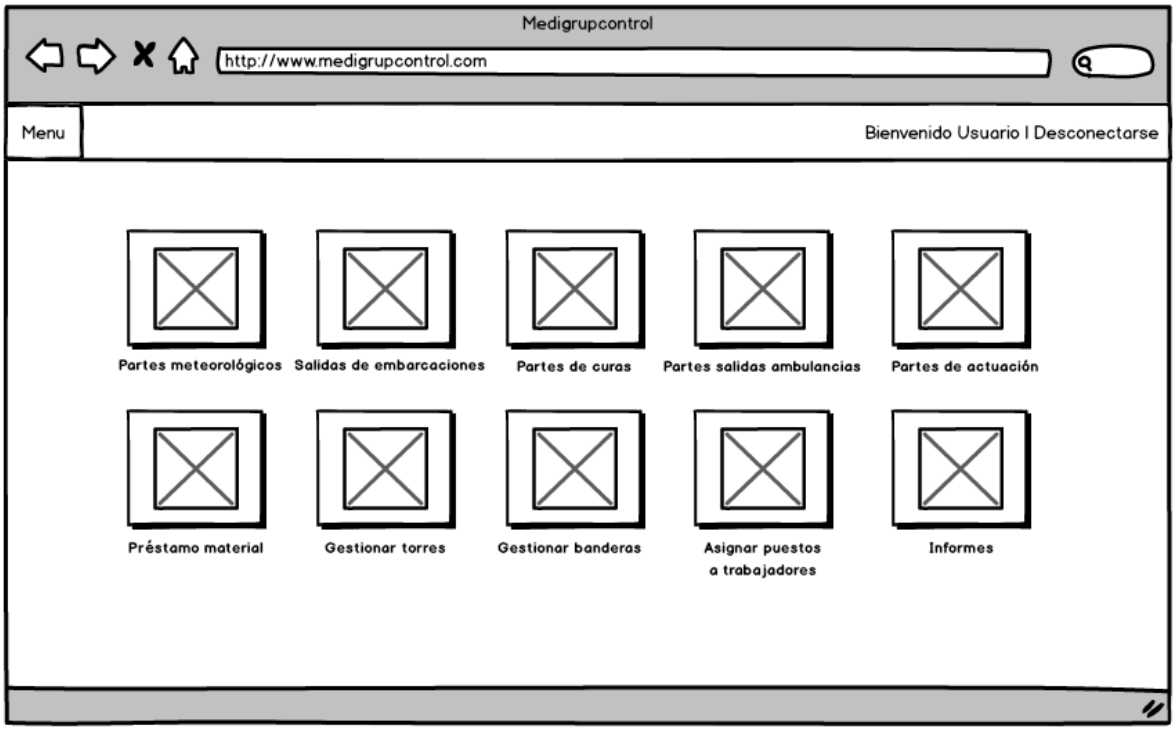

Figura 5.6: Prototipo. Menú principal.

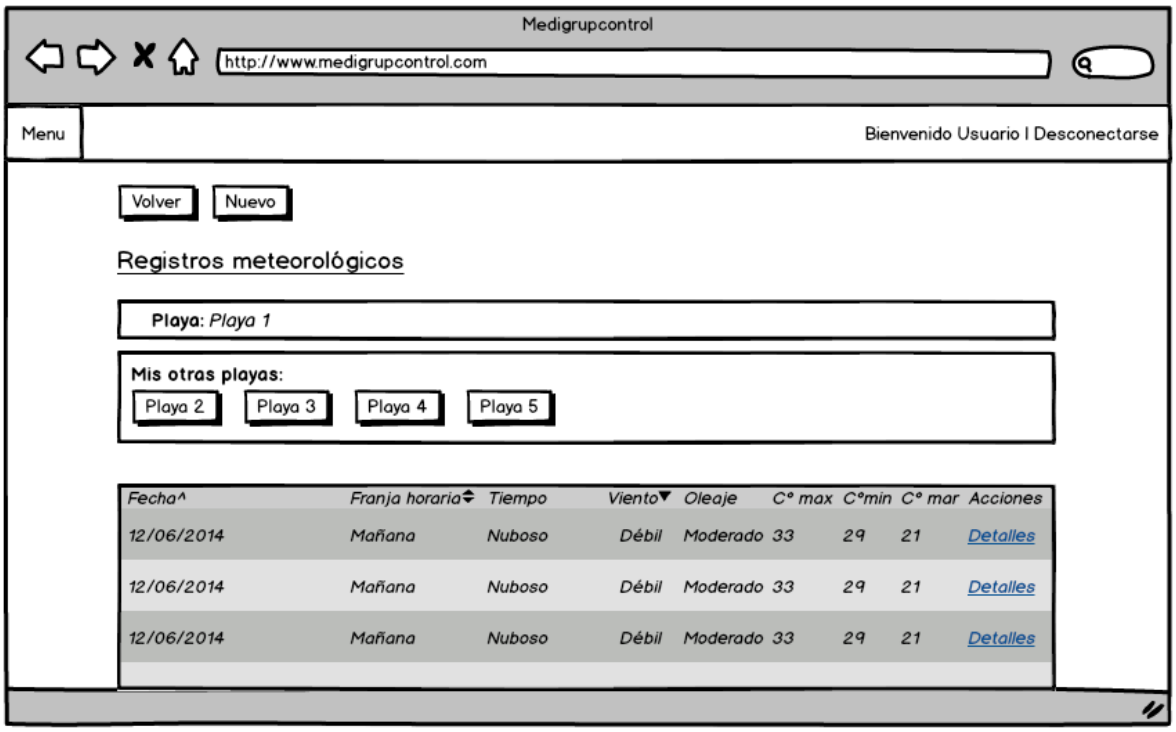

Figura 5.7: Prototipo. Partes meteorológicos.

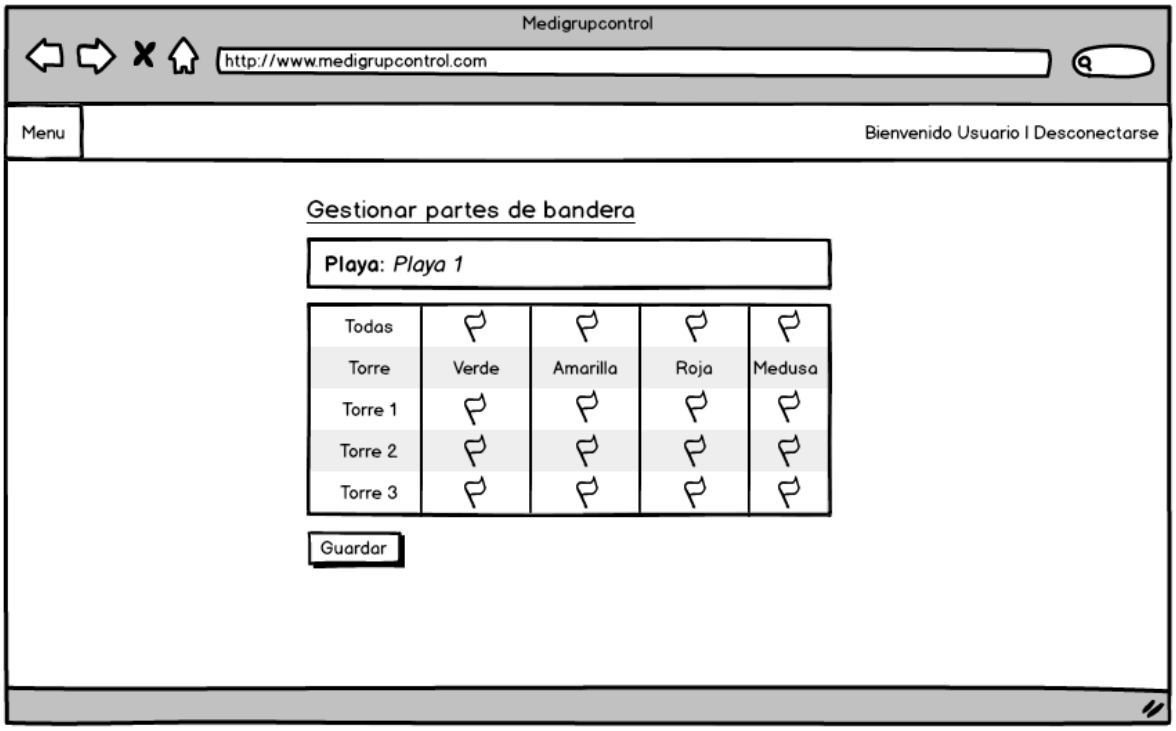

Figura 5.8: Prototipo. Registro de banderas.

Capítulo 6

Implementación

En este capítulo se explica el proceso llevado a cabo para la elaboración del producto final del proyecto. Esta fase es donde se refleja cómo de bien se han estimado los tiempos, costes y recursos necesarios para el proyecto. También se comprueba si el producto diseñado es el que realmente necesita el cliente, o por el contrario, se ha invertido tiempo y recursos en un producto carente de utilidad.

6.1. Detalles de implementación

Aquí podemos encontrar los detalles de la implementación del proyecto. Se detalla el uso y configuración del generador Gii (Apartado 6.1.2), se explican la creación y modificación de formularios simples (Apartado 6.1.3) y complejos (Apartado 6.1.4), la creación de roles y sus permisos correspondientes (Apartado 6.1.5), y el generador de informes en hojas de cálculo (Apartado 6.1.6).

6.1.1. Funcionamiento de Yii

Antes de comenzar a utilizar las partes, fué necesario realizar pruebas para comenzar a comprender el uso del *framework*. Para comenzar, se comprendió cómo funciona el *framework* por dentro.

De forma interna, Yii utiliza el patrón arquitectónico Modelo-Vista-Controlador (MVC). Este patrón fue inventado por un desarrollador del lenguaje Smalltalk llamado Trygve Reensauk [8] y permite independizar el modelo de datos de la vista. Esto le da la capacidad a ambas partes de ser separables y fácilmente reemplazables. Dado que este patrón sirve como guía y no es estricto en su definición de flujo de datos, la implementación por parte de Yii sigue el flujo mostrado en la Figura 6.1. En ella se puede ver como el modelo, la vista y el controlador se intercomunican todos entre sí, mientras que existen componentes añadidos como los *widgets* y un objeto de sesión *application*.

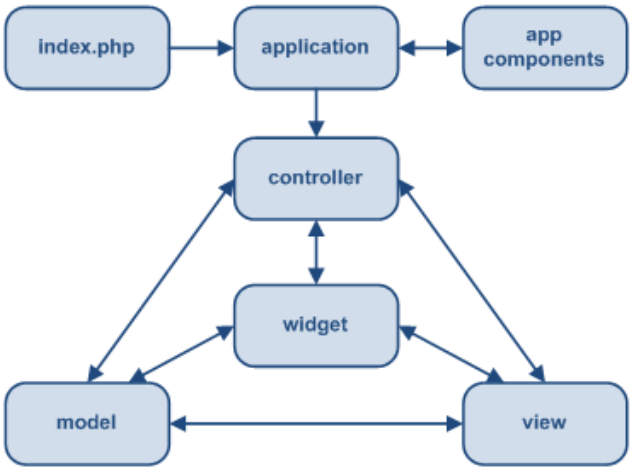

Figura 6.1: Estructura de una aplicación Yii.

Dentro de esta estructura, las tareas y peticiones realizadas a la aplicación se resuelven de la siguiente manera (Figura 6.2). En primer lugar, se crea una nueva instancia de la aplicación, inicializando las funcionalidades esenciales. Una vez ejecutada la instancia, ésta analiza la petición recibida y decide a qué controlador le corresponde procesarla con la ayuda del componente *urlManager*. Una vez se conoce el controlador a utilizar, se instancia un objeto del mismo. Con la instancia del controlador, se busca la acción que se desea ejecutar. En los controladores, las acciones que se pueden realizar vienen precedidos por el prefijo *action* y se pueden definir permisos para todas las acciones. Una vez ejecutada la acción, se hace uso del modelo necesario y se crea la vista correspondiente al resultado. La vista utiliza los *widgets* necesarios y crea el *layout* correspondiente.

Para la creación del layout, Yii utiliza un flujo de vistas en el que va incluyendo una dentro de otra (Figura 6.3). Mediante el uso de *renderPartial*, Yii va introduciendo cada elemento dentro de su plantilla superior. Este segmento de vista se procesa y posteriormente se almacena en una variable llamada $content. Esta variable va pasándose hacia su plantilla superior, pasando por la vista inicial, la plantilla elegida de 1 o 2 columnas y finalmente a la plantilla general de la aplicación. Este sistema permite unificar las partes comunes de las vistas y simplificar el desarrollo de una aplicación.

6.1.2. El generador Gii

El Yii Framework viene con un generador de código integrado llamado *Gii*. Este generador de código tiene una interfaz web, la cual es accesible al crear un nuevo proyecto Yii. Existe un fichero de configuración llamado *main.php* que se debe modificar la primera vez que se quiere utilizar un nuevo proyecto. En él establecemos la URL y las credenciales para acceder a la base de datos que tengamos instalada. Este framework es compatible con una gran cantidad de sistemas gestores de bases de datos, entre ellos el utilizado en este proyecto, MySQL.

Una vez configurado este fichero, se puede acceder a la interfaz web de generación de código

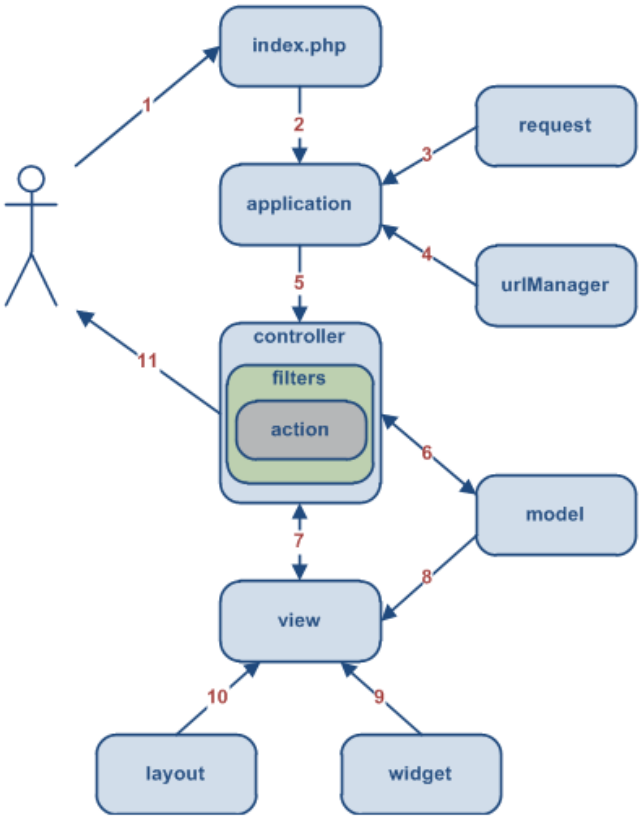

Figura 6.2: Flujo de una petición en Yii.

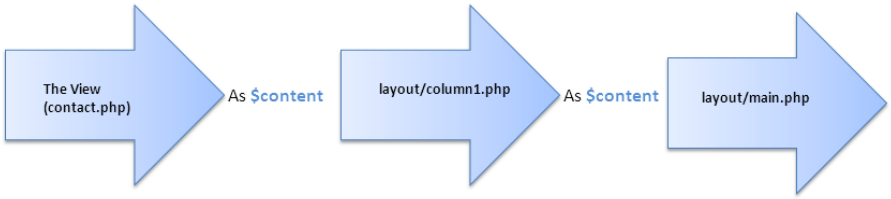

Figura 6.3: Flujo de procesamiento de vistas en Yii.

(Figura 6.4). Lo que hace especial a este generador de código, es que automáticamente enlaza las tablas de la base de datos conectada con la interfaz web, y genera todo el CRUD necesario para cualquier aplicación de gestión. En esta generación se incluyen las interfaces gráficas web, las cuales se pueden modificar en el generador antes de crearlas. Si se modifican las plantillas del generador Gii, el tiempo de desarrollo se disminuye considerablemente, sobre todo para aplicaciones grandes de gestión. Con modificar una sola plantilla, el generador es capaz de extrapolarla a todas las clases que se generen.

El generador de código usa el mismo lenguaje que el resto del framework, PHP. Desde las plantillas se imprime o deja fuera de las etiquetas de PHP el código que se desea que se genere. Dicho de otro modo, el generador es una serie de ficheros PHP que generan otros ficheros PHP o Html. En la Figura 6.5 se puede ver un ejemplo. Se puede ver que se utilizan métodos del

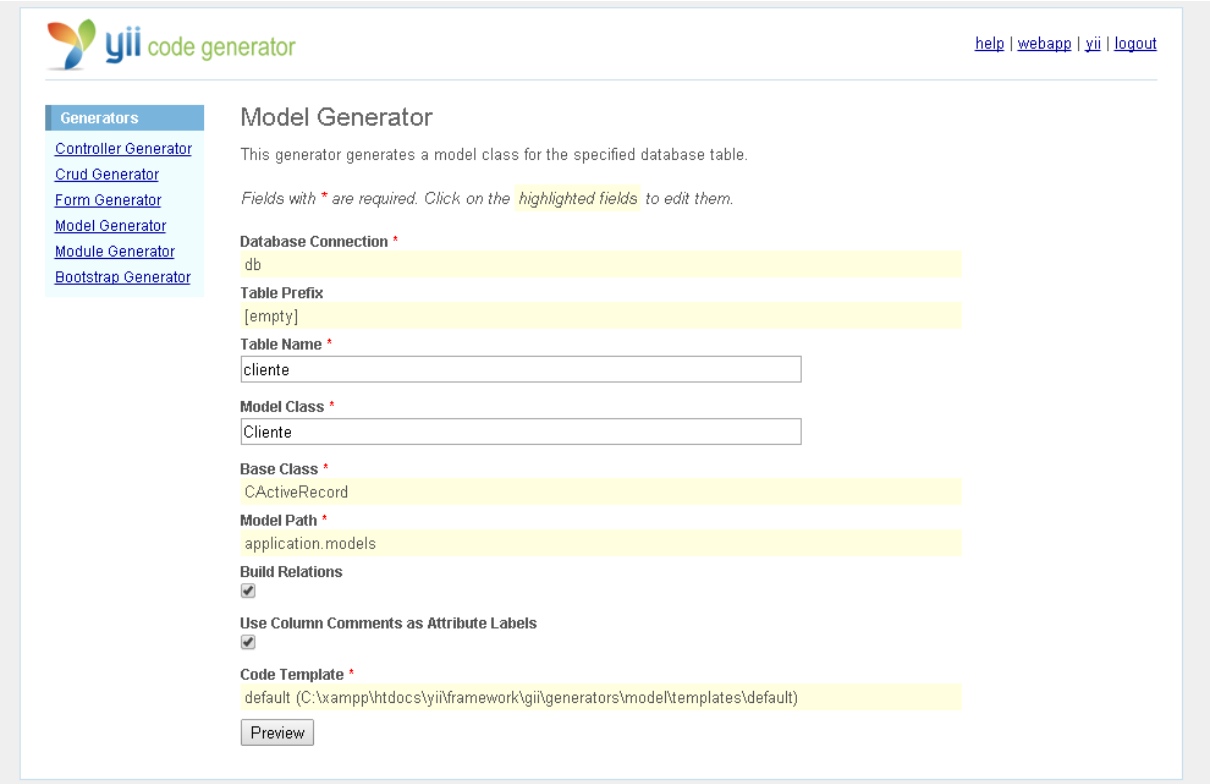

Figura 6.4: Generador Gii. Creación de modelo.

propio generador, como en la línea 38 el método *class2name()* que devuelve el nombre de la clase a generar. Estos métodos son los que permiten la creación genérica de las clases y vistas.

```
38 <h1>Gestionar <?php echo $this->pluralize($this->class2name($this->
       modelClass)); ?></h1>
39
40 <?php echo "<?php␣echo␣CHtml::link('Búsqueda␣avanzada',␣'#',␣array('class'␣
       =>␣'search-button'));␣?>"; ?>
41
42 <div class="search-form" style="display:none">
43 <?php echo "<?php␣\$this->renderPartial('_search',␣array('model'=>\$model,)
       );?>\n"; ?>
44 </div>
```

Figura 6.5: Generador Gii. Trozo de código del generador.

El generador por defecto, dada una tabla de la base de datos, genera las clases que se ven en la Figura 6.6. En el controlador existen métodos para rellenar todas las vistas creadas, permisos por defecto para usuarios genéricos de la aplicación, y la funcionalidad CRUD completa de cada entidad. El modelo recoge todas las restricciones de la base de datos e implementa validadores genéricos para el formulario. Por ejemplo, digamos que un campo de la base de datos es un *INT (50) NOT NULL*. El modelo contendrá un validador en el formulario para comprobar si se ha enviado algo (not null), si es un número y si tiene un máximo de cincuenta caracteres.

Pero existe un problema con este generador. Lo que se genera automáticamente esta muy

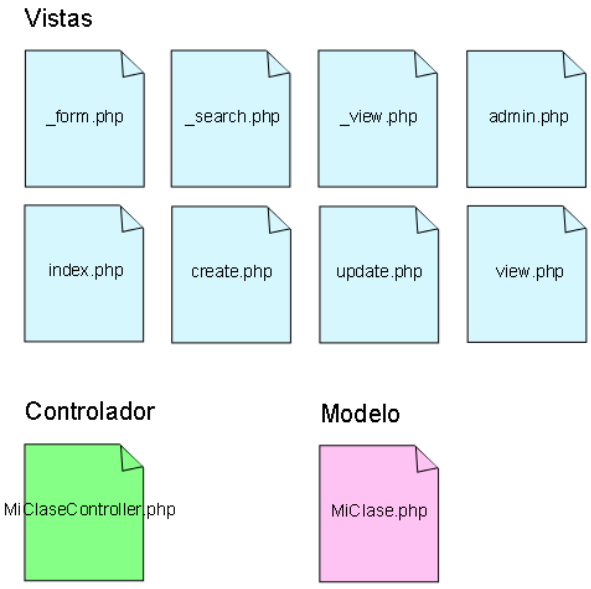

Figura 6.6: Generador Gii. Clases generadas de MiClase.

bien para comenzar un proyecto, evita tener que arrancar de cero. Pero todo requiere una gran cantidad de modificaciones para acomodarse a las necesidades del proyecto, comenzando por el aspecto. Para que el generador generase las vistas con la biblioteca de diseño adaptable Bootstrap, fué necesario modificar cada una de las plantillas del generador. Utilizando la extensión *Yiistrap* se cambiaron los *widgets* por defecto por los que implementan esta biblioteca. Se añadieron todas las clases CSS necesarias a los elementos que lo requerían y se adaptó la estructura general de todas las vistas a diseño adaptable.

Aparte de las vistas que se generan con este generador, existe una serie de *templates* por defecto que definen el número de columnas, el menú superior y el pie de todas las vistas. Estos *templates* no se generan con Gii, sino que se tienen al comienzo y se modifica lo requerido una sola vez. Se optó por utilizar un menú Bootstrap utilizando unas animaciones con *jQuery* que funciona perfectamente en dispositivos móviles.

Para simplificar la tarea de realizar las vistas a largo plazo, se dedicó una gran cantidad de tiempo a modificar el generador para conseguir el aspecto deseado de generación automática. Este tiempo invertido, se compensó con el ahorro de tiempo necesario para crear cada vista y simplificó mucho las tareas posteriores en general. Se añadieron una serie de elementos que por defecto no aparecían, como es el caso de una tabla de administración con búsqueda mediante AJAX. Esta tabla funciona perfectamente de forma adaptable y su diseño es mucho mas atractivo que el de la tabla original (Figura 6.7).

El generador se utilizó solamente en el entorno de pruebas por motivos de seguridad. Si alguien con intenciones nocivas accediese al generador, podría tener unas consecuencias atroces para la aplicación. Por ello, se eliminó del entorno real, dejando sólo el contenido generado previamente.

Figura 6.7: Generador Gii. Comparativa de tablas generadas.

6.1.3. Formularios básicos

Una vez generados los formularios con Gii, fué necesario realizar una gran cantidad de modificaciones manuales. Además de ello, la funcionalidad del controlador y el modelo no funcionan como se desea nada más ser generados. Por ello, se comenzó modificando un formulario básico: el registro de usuarios. Este formulario, al estar destinado a ser utilizado en ordenador de sobremesa y no en dispositivos móviles, no era necesario que fuese estrictamente adaptable.

Lo primero que se tuvo que hacer al comenzar a editar el formulario, fué quitar los campos innecesarios y renombrar los existentes a lo deseado. Por ejemplo, habría que quitar el campo *ID* ya que este campo no debe ser visible para un usuario. Tampoco es necesario que vea el campo de fecha de registro, ya que es un campo privado de consulta.

Tras esto, se procedió a ajustar los campos al tamaño deseado, el tipo adecuado y comenzar a configurar los filtros. El *framework* incorpora una serie de filtros por defecto y la posibilidad de crear propios. Estos filtros funcionan validándose mediante Javascript (únicamente en el cliente), mediante AJAX (se comprueba en el servidor en tiempo real y se informa al cliente) y directamente en el servidor antes de almacenar los datos. Es importante que se validen los formularios tanto en el servidor como en el cliente. En el cliente es importante para tener una rápida respuesta y para quitar carga al servidor. En el servidor es importante para asegurarse que lo que se va a almacenar no contiene nada no permitido.

Para ciertos campos, como la fecha de nacimiento, se optó por utilizar *widgets*, como el *datepicker*. Esto permite seleccionar la fecha de forma mucho más sencilla, además de asegurar que el formato insertado es el deseado por la aplicación. Estos *widgets* vienen incluidos en la extensión Yiistrap. Para crear una contraseña, se añadió la opción de generar automáticamente una. Para conocerla, al insertar el correo electrónico, se envía la contraseña generada directamente al completar el registro.

Las funcionalidades en los controladores y modelos incluidos por defecto son escasas y requieren modificaciones. Las funcionalidades básicas CRUD necesitan incorporarse con las entidades que tengan relación. Por ejemplo, los formularios de salida de ambulancia requieren estar relacio-

nados con los de actuación y los de curas en algunas ocasiones. Esta funcionalidad requiere crear una transacción para asegurarse que los datos se guarden de forma consistente y no ambigua.

6.1.4. Formularios complejos

Los formularios creados para los partes son más complejos. Muchos tienen relaciones entre sí, y para ello fué necesario investigar acerca del uso de relaciones y sus restricciones en Yii. Por defecto, Yii utiliza el método de recogida de datos de la base de datos implementando el patrón *lazy loading*. Este método, en contraposición a otro método llamado *eager loading*, carga los datos únicamente cuando son solicitados. Al existir tanta cantidad de datos, es importante que únicamente se carguen los datos que se requieren en el momento, mejorando mucho el uso de memoria temporal.

En los formularios de gestión se encontraron algunas dificultades. Una de estas fué las búsquedas en la base de datos. La tabla que se genera para mostrar los datos tiene por defecto campos de búsqueda asociados a éste. Para utilizarse, se ha de implementar una petición AJAX que solicite los datos filtrados por el elemento introducido. Para utilizar estas peticiones, se tuvo que averiguar como asegurarse de que la petición fuese mediante AJAX, además de comprobar la validez de la petición. Los métodos de búsqueda avanzada no hacían uso de peticiones AJAx, ya que éstos envían todos los datos y recargan la página. Para realizar el filtrado por fechas, se optó por separar la fecha en día, mes y año para poder tener una mayor flexibilidad en las búsquedas.

A la hora de realizar los partes de banderas, al ser una tarea repetitiva que se iba a realizar a diario, se optó por crear un *widget* utilizando HTML, CSS y javascript. Este *widget* consiste en un selector que muestra las torres disponibles en la playa seleccionada, junto con los tipos de banderas disponibles. Permite seleccionar la misma bandera para todas las torres con solo un botón, permitiendo realizarse de forma sencilla desde tabletas. Para almacenar los datos pulsados, se hizo uso del atributo *data* de HTML5. Este atributo permite almacenar cualquier valor que se desee en un elemento HTML utilizando el prefijo *data-*. Con ello, resulta mucho más sencillo saber qué elemento ha sido seleccionado sin tener que utilizar campos ocultos u otros métodos menos óptimos.

Otro reto en los formularios, fué la asignación de tripulantes a una embarcación. Cuando se crea un parte de salida de embarcación, es necesario asignar la gente que ha estado en ella, y puede ser más de uno. Por ello, se modificó un *widget* de Bootstrap llamado Selector el cual permite buscar en un combobox tecleando. Una vez se encontraba el tripulante seleccionado, se pulsa añadir y se lista con los demás. Al lado de cada tripulante añadido sale un botón para eliminarse. Este *widget* se creó utilizando javascript para ir añadiendo o borrando de una lista, así como comprobando que no estuviese añadido previamente. Por ello, se tenía un índice con los identificadores de cada uno de los tripulantes añadidos.

El formulario de recuperación de contraseña también tuvo algunas complejidades. Para que no se realizasen peticiones no gratas al sistema de recuperación de contraseña a través de *bots*, se añadió un Captcha[1]. Esto asegura que sea un humano el que realiza la petición. Se introduce el correo electrónico y, en caso de existir en el sistema, se envía un correo para

[1]Completely Automated Public Turing test to tell Computers and Humans Apart.

recuperar la contraseña. Para enviar el correo se utilizó la biblioteca PHPMail. Ésta permite configurarse para enviar a través de PHP Pear correos electrónicos. Los usuarios que olvidasen su contraseña, reciben una nueva generada previamente por un sistema automático. Si lo desean pueden cambiarla luego accediendo a su cuenta.

6.1.5. Roles y permisos

Para gestionar los permisos de la aplicación, se crearon una serie de roles que definen el acceso o denegación a las diferentes partes de la aplicación. Estos roles fueron diseñados teniendo en cuenta los trabajadores que iban a utilizar la aplicación. Estos roles corresponden con los actores definidos en la Tabla 4.1.

Para implementar estos permisos, se implementó un *Role-Based Access Control (RBAC)* extendiendo el componente CWebUser de Yii. Este componente está pensado para realizar una autenticación de usuario estándar, y está enlazado con los permisos independientes de cada controlador para las funcionalidades de los mismos. Para ello, cuando se realiza una autenticación por parte del usuario, se almacena su identificador en sesión. Este identificador se utiliza posteriormente para realizar una comprobación de su rol y permitir o denegar el acceso a la funcionalidad según proceda.

Esta forma de gestionar los permisos simplifica mucho la tarea, ya que desde un único punto del controlador es posible gestionar los permisos de cada método disponible. Se realiza en un método *accessRules()*, y permite definir una gran cantidad de aspectos, como el tipo de petición HTTP que se acepta (GET, POST, PUT o DELETE, y si se acepta mediante AJAX o no). Es posible abstraer todos estos permisos a una clase que los centralice, pero debido a que había una gran cantidad de métodos se optó por tenerlos separados según sus clases. Se puede ver un ejemplo de una configuración de acceso para ciertos roles en la Figura 6.8. Se creó un método específico para comprobar qué roles tienen permisos sin tener que comprobarlos uno a uno.

```
1  public function accessRules() {
2      return array(
3          array('allow',//socorrista puede ver/crear/actualizar
4              'actions'=>array('view', 'create', 'update'),
5              'expression'=>'Yii::app()->user->checkAccess("socorrista")'
6          ),
7          array('allow',//admin y jefe de playa pueden borrar/administrar
8              'actions'=>array('admin','delete'),
9              'expression'=>'Yii::app()->user->checkAccess(array("admin",␣"
                   jefe_playa"))'
10         ),
11         array('deny',
12             'users'=>array('*'),//Denegamos acceso al resto de usuarios.
13         ),
14     );
15 }
```

Figura 6.8: Controlador. Control de acceso.

6.1.6. Generador de informes

Uno de los requisitos solicitados por el cliente, era que se pudiesen generar informes de los eventos relacionados con las playas. Estos informes sirven para tener un seguimiento de todo lo que va ocurriendo en ellas, además de ser un requisito necesario para los ayuntamientos. La empresa encargada de gestionar estas playas, debe realizar informes cada cierto tiempo que sigan un formato especificado por ellos y entregarlo.

Para ello se implementó un apartado que permite generar informes en formato de hoja de cálculo. Estos informes son totalmente configurables, y permiten seleccionar un rango de fechas, los usuarios que se incluirán, el autor del informe, las playas involucradas e incluso los campos a mostrar de cada tipo de partes. Esto permite, por ejemplo, ver el número de picaduras de pez araña ocurridas en la segunda quincena del mes de julio en una playa en particular.

A la hora de desarrollar este generador, se buscó un módulo que fuese capaz de permitir una generación de informes en el formato de hoja de cálculo solicitado (Microsoft Excel 2007, .xlsx). Al final se optó por utilizar PHPExcel. Este proyecto proporciona los elementos necesarios para generar informes en una variedad de formatos y utiliza el estándar Microsoft OpenXML. Además es fácil de integrar como un componente de Yii.

El uso de este componente era un poco complejo, y dado que se debía seguir un formato estricto, se dedicó mucho tiempo a calcar el diseño exacto.

6.2. Seguridad

Una aplicación de gestión empresarial, al tratar con datos relativos a ella, debe procurar ser lo más segura y robusta posible. Por ello, a la hora de configurar y desplegar la aplicación en el entorno real, se aseguraron algunos puntos clave del *framework* en temas de seguridad. Estas medidas y configuraciones permiten tener una protección básica de la aplicación. Para proteger la aplicación en un mayor nivel, lo ideal sería realizar una auditoría de seguridad en la que se encontrasen las vulnerabilidades a las que se encuentra expuesta y cubrirlas. Las medidas de seguridad que se tuvieron para proteger la aplicación pretendieron cubrir algunas de las vulnerabilidades de aplicaciones web más extendidas y utilizadas:

- **Inyección SQL.** Esta vulnerabilidad es la más típica y extendida en las aplicaciones web actuales [11]. Consiste en introducir una sentencia SQL maliciosa en el transcurso de una consulta legítima de la aplicación al recibir variables del usuario. A pesar de ser muy grave, es fácilmente evitable si se filtran las variables a utilizar en una consulta SQL antes de utilizarse. Para este caso, lo ideal es usar *prepared statement*. Es una sentencia parametrizada que permite solamente introducir parámetros y no otras sentencias, ya que se filtran de antemano. Esto evita por completo esta vulnerabilidad, y ha sido utilizado a lo largo de la aplicación.
- **Cross-site Scripting (XSS).** Esta vulnerabilidad permite introducir código malicioso en la aplicación para que otros usuarios lo usen sin saberlo y recoger datos personales. Para prevenir esto, Yii incorpora un filtro que utiliza el HTML Purifier [23]. Este purificador

se encarga de filtrar el contenido insertado por los usuarios de la aplicación, asegurándose de que no contiene código malicioso.

- **Cross-site Request Forgery (CSRF).** Esta vulnerabilidad hace que cuando un usuario entra en una página web maliciosa, ésta cause que el navegador ejecute una acción no deseada en una página que sí es fiable. Para evitar este ataque, es importante tener un reconocimiento de que las peticiones enviadas son legítimas. Por ello, se ha activado un *token* generado en base a la *cookie* que se comprueba en el lado del servidor para cada petición. Esto ayuda a reconocer con facilidad si la petición es legítima o no.
- **Protección de *cookies*.** Las *cookies* almacenadas en el navegador son una gran vulnerabilidad para cualquier aplicación web, ya que éstas tienen el ID de la sesión que se está utilizando en él. Este ID puede dar acceso a otros datos confidenciales e incluso al uso de la sesión. Para protegerlas, Yii implementa un validador de *cookies*, comprobando el HMAC[2] de sus valores para asegurar que son los correctos.
- **Magic URLs.** Esta vulnerabilidad permite que, mediante el cambio de los valores GET de una petición, se ejecuten acciones que originalmente no estaban destinadas para un usuario en particular. Para ello, aparte de realizar autenticación basada en roles, se comprueba que cada acción realizada por un usuario esté dentro de sus privilegios particulares.

A pesar de tener cubiertas estas vulnerabilidades, existe una gran cantidad de aspectos que pueden fallar a la hora de proteger una aplicación de personas no gratas. Por ello, además de cubrir estas vulnerabilidades, se han tomado otras medidas de seguridad para proteger aún mas la aplicación:

- **Proteger contraseñas.** Las contraseñas almacenadas se encriptan automáticamente con el algoritmo SHA-256[3]. Esto permite que, aunque se accediesen a los datos sensibles de la aplicación, se extraerían de forma encriptada.
- **Archivo *.htaccess* de Apache.** Se configuró el archivo de configuración distribuida o *hypertext access* del servidor para bloquear la ejecución de archivos PHP en la carpeta de recursos estáticos (assets/).
- **Cambiar carpeta *protected* de ubicación.** A pesar de estar protegida por el archivo de configuración distribuida del servidor, es recomendable sacarla directamente de la carpeta del proyecto.
- **Configurar archivo *php.ini*.** Este archivo es donde se declaran las directivas que debe seguir la instancia de PHP del servidor. Para mayor seguridad, se desactivaron algunas variables que podían ser un riesgo de seguridad.

6.3. Pruebas

A la hora de realizar un proyecto de dimensiones considerables, es de especial importancia utilizar un sistema de pruebas que facilite la depuración del sistema. Durante la implementación

[2]Hash-based Message Authentication Code.
[3]Secure Hash Algorithm.

se pueden utilizar herramientas que faciliten la depuración de código, como en este caso un inspector de elementos web y un IDE con *breakpoints*. A largo plazo, lo ideal es realizar pruebas unitarias en conjunto con el proyecto. Esto puede resultar tedioso al principio, ya que requiere una mayor cantidad de tiempo invertido para realizar una funcionalidad de forma inicial, pero esto se compensa a largo plazo evitando fallos futuros. En este proyecto no se ha hecho uso de pruebas unitarias, pero se han utilizado otros tipos de pruebas explicados a continuación.

6.3.1. Pruebas en tabletas

Una vez finaliza la aplicación se realizaron pruebas de todas las funcionalidades de la aplicación por parte de los empleados de la empresa de forma previa a la entrega del producto al cliente. Para estas pruebas, se utilizaron las tabletas que se iban a entregar posteriormente a los usuarios de la aplicación para su uso. A pesar de entregarse en forma de versión alfa, se aseguró que no hubieran grandes fallos en la aplicación. Para ello cada usuario realizó las siguientes acciones a modo de prueba:

- Registrarse como usuario de la aplicación. Se registraron todos los roles posibles desde la cuenta de administrador. Esta aplicación no pretende registrar usuarios a distancia, por tanto el registro de usuarios únicamente está disponible desde el panel del administrador.
- Autenticarse y desconectarse. Se comprobó que todos los usuarios podían entrar y salir sin problemas, independientemente del rol.
- Asignar playas. Se asignaron playas a los usuarios y se comprobó que se listasen y estuviesen disponibles para su selección.
- Cambiar contraseña. Los usuarios se cambiaron la contraseña y comprobaron que funcionaba todo correctamente. Se probó tanto desde "He olvidado mi contraseña", como desde la opción "Cambiar mi contraseña" del menú de usuario.
- Acciones específicas de cada rol. Se comprobó que cada rol pudiese realizar únicamente las acciones asociadas a su rol. Para ello se crearon partes de todo tipo, préstamos de material, gestión de torres y generación de informes.

6.3.2. Registro de eventos

Para realizar un seguimiento de los errores críticos de la aplicación, se implementó un sistema de registro de eventos. Este registro guarda en la base de datos todas las excepciones lanzadas por la aplicación, sean críticos o no, e información relevante como el usuario que lo ha producido, la fecha y hora y otros detalles. La tabla necesaria para realizar esto estaba pensada desde el principio y se incluye en el modelo conceptual de datos. Esta acción también se incluye como uno de los casos de uso de la aplicación, ya que estaba pensado como una funcionalidad del administrador.

Este seguimiento ha permitido encontrar una gran variedad de errores reportados, de los que no se sabían apenas datos. Gracias a los detalles de la excepción y al usuario y fecha se

ha sabido identificar de qué errores hablaban los usuarios. También existía una serie de fallos menores que ha sido posible depurar gracias a esta herramienta. Cabe destacar que el Yii framework implementa una versión básica de esto llamado CDbLogRoute, el cuál fue extendido y al que se le añadió una interfaz gráfica para facilitar su uso.

El nivel de excepciones puede variarse directamente desde la configuración de Yii. Esta configuración se aplica a los niveles de muestra de error de PHP. Es importante que el usuario no pueda ver estos errores, sino que le aparezcan pantallas con códigos de error HTTP como el 500 en caso de error interno del servidor.

6.4. Reuniones

A la hora de crear la aplicación, al ser tan cercano el cliente, se probó una versión alfa directamente con los usuarios finales. Estos usuarios se reunían periódicamente con la empresa para dar retroalimentación de posibles defectos o fallas que encontrasen en la aplicación, así como ideas para mejorarla. Esta retroalimentación sirvió de mucha ayuda para crear la aplicación a medida para estos usuarios, además de servir como un buen aprendizaje de los gustos en general de los usuarios. Se realizó un total de tres reuniones. A continuación se resume lo ocurrido en cada una de ellas.

En la primera reunión, fueron viniendo todos los usuarios de la aplicación para mostrarles una pequeña demostración del uso de la misma. En esta reunión expusieron dudas y sugerencias iniciales, como la relación entre los partes de incidencias. También se registraron uno por uno en la aplicación, y se descubrieron unos cuantos detalles que no eran obvios inicialmente, pero que tras esto se implementaron.

Se notó que los usuarios, utilizando el *datepicker*, recorrían uno por uno los años hacia atrás hasta llegar al suyo. Al estar puesto inicialmente en la fecha actual, era una tarea muy costosa. Por ello se optó por cambiar el seleccionador para que comenzase en una vista de años y no meses. Esto permite seleccionar la fecha de nacimiento de una forma mucho más rápida.

Otro detalle descubierto fué que la gente introducía en el campo dirección todos los datos, incluida la provincia. Luego llegaban al campo provincia y tenían que subir al campo dirección para borrarla. Para solventar esto, se cambiaron de lugar ambos campos, para que la provincia apareciese primero. También se puso un pequeño mensaje a modo de ayuda para que la gente supiese los datos necesarios del campo dirección.

El último detalle descubierto en esa reunión fué que la gente no sabía si el DNI o NIE incluía letra o no. Por ello se añadió un mensaje a modo de ayuda con el formato que se deseaba de DNI y NIE.

En las siguientes dos reuniones, se comentaron pequeños *bugs* encontrados durante el uso de la aplicación. Estas dos reuniones fueron bastante breves, y era para comentar algo que viesen que fallaba o alguna propuesta de mejora. Por lo general, se mostraron satisfechos con las funcionalidades que ofrecía la aplicación y con el ahorro de tiempo que involucraba respecto a su sistema antiguo.

6.5. Documentación

Para este proyecto, por petición del cliente, era necesario realizar una ayuda interna para la aplicación. Ésta recoge todos los tipos de usuarios y sus descripciones. También contiene de forma detallada los pasos a seguir para realizar cada una de las acciones generales de las que dispone la aplicación.

Para la realización de esta ayuda, se implementó un menú desplegable que simplifica la navegación del mismo. El resultado final de esta ayuda puede encontrarse tanto en la aplicación final puesta en producción, como en el anexo B adjunto.

Capítulo 7

Conclusiones

A lo largo de la estancia en prácticas y el consecuente desarrollo del proyecto, he adquirido una gran cantidad de conocimientos y experiencias. Para empezar, he aprendido mucho sobre el lenguaje PHP. Este lenguaje está muy solicitado hoy en día en el mundo laboral, y me ha permitido ampliar mis conocimientos generales. Por otra parte, integrarse en un equipo de desarrollo me ha permitido observar las diferentes formas de trabajar de un grupo multidisciplinar.

Crear una interfaz adaptable preparada para usarse en tabletas me ha aportado bastantes conocimientos aplicables a las aplicaciones requeridas hoy en día. Es muy importante que toda aplicación sea adaptable a varios tipos de resoluciones ya que cada vez hay una mayor cantidad de usuarios que acceden a ellas a través de dispositivos móviles.

Al comenzar el proyecto, también observé que cada tecnología nueva que se pretende utilizar requiere un tiempo de formación y práctica para poder usarse de forma correcta. He notado mi evolución en cuanto a conocimientos y soltura a lo largo de la duración de la estancia, y esto me ha permitido comprobar que los estudiantes estamos más preparados para el mundo laboral de lo que a priori podría pensar.

El tema del proyecto me ha gustado mucho, sobre todo sabiendo que iba a ser utilizado por usuarios que realmente lo necesitan en cuanto estuviese acabado. Esta experiencia sí que es algo que durante el tiempo de estudio en la universidad se echa en falta. Da mucha satisfacción saber que el proyecto realizado está siendo útil para un conjunto de usuarios. El hecho de haber tenido reuniones con los usuarios, y haber recibido retroalimentación directa también me ha animado mucho a mejorar la aplicación y cumplir los requisitos que esperan.

La integración del proyecto con otras partes desarrolladas por el resto del equipo me hizo aprender mucho acerca de cómo acomodar código escrito por diferentes personas entre sí. Es sorprendente la cantidad de formas diferentes en las que se puede desarrollar una misma funcionalidad según el programador.

El proyecto ha tenido una gran aceptación por parte de los usuarios. Ha sido utilizado en las playas de la Comunidad Valenciana durante el verano, y he recibido gran satisfacción al ver el uso y comentarios recibidos.

Este proyecto, al pertenecer a una gran aplicación de gestión de todo tipo de centros, seguirá ampliándose poco a poco hasta cubrir todas las necesidades que requiere la empresa. Además, los conocimientos adquiridos son extrapolables a prácticamente cualquier aplicación de gestión que se requiera, lo cual me ayuda a tener mayor confianza a la hora de afrontar proyectos de ese ámbito.

Bibliografía

[1] 2amigOS!. Bootstrap Yii extension. http://www.getyiistrap.com/. [Consulta: 15 de Septiembre de 2014].

[2] 2amigOS!. Foundation Yii extension. http://yiifoundation.2amigos.us/. [Consulta: 15 de Septiembre de 2014].

[3] 2amigOS! Consulting Group. 2amigOS!. http://2amigos.us/. [Consulta: 21 de Septiembre de 2014].

[4] Apple Inc. Safari Web Browser. https://www.apple.com/es/safari/. [Consulta: 17 de Septiembre de 2014].

[5] John Bertucci. Design for developers: Bootstrap 161. http://es.slideshare.net/jmbertucci/design-for-developers-introduction-to-bootstrap-3. [Consulta: 16 de Septiembre de 2014].

[6] Will Bond. Package Control. https://sublime.wbond.net/. [Consulta: 16 de Septiembre de 2014].

[7] Bootstrap. Bootstrap version 2.3.2. http://getbootstrap.com/2.3.2/. [Consulta: 15 de Septiembre de 2014].

[8] Steve Burbeck. How to use Model-View-Controller (MVC). https://www.owasp.org/index.php/SQL_Injection. [Consulta: 19 de Septiembre de 2014].

[9] Creative Commons. Creative Commons Attribution 3.0 Unported. http://creativecommons.org/licenses/by/3.0/. [Consulta: 21 de Septiembre de 2014].

[10] Fog Creek Software, Inc. Fog Creek Software. http://www.fogcreek.com/. [Consulta: 15 de Septiembre de 2014].

[11] OWASP Foundation. SQL Injection. https://www.owasp.org/index.php/SQL_Injection. [Consulta: 21 de Septiembre de 2014].

[12] The Apache Software Foundation. Apache Software License Version 2. http://www.apache.org/licenses/LICENSE-2.0. [Consulta: 20 de Septiembre de 2014].

[13] Free Software Foundation, Inc. GNU General Public License. http://www.gnu.org/copyleft/gpl.html. [Consulta: 16 de Septiembre de 2014].

[14] Dave Gandy. Font Awesome Icons. http://fortawesome.github.io/Font-Awesome/. [Consulta: 21 de Septiembre de 2014].

[15] Google. Chrome Web Browser. http://www.google.com/intl/es_es/chrome/. [Consulta: 17 de Septiembre de 2014].

[16] EMCA International. ECMA Script specifications. http://www.ecma-international.org/ecma-262/5.1/. [Consulta: 15 de Septiembre de 2014].

[17] SIL International. SIL Open Font License (OFL). http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&id=OFL. [Consulta: 16 de Septiembre de 2014].

[18] Microsoft Corporation. Internet Explorer. http://windows.microsoft.com/es-es/internet-explorer/download-ie. [Consulta: 17 de Septiembre de 2014].

[19] Mozilla. Mozilla Firefox. https://www.mozilla.org/es-ES/firefox/new/. [Consulta: 17 de Septiembre de 2014].

[20] Open Source Initiative. The MIT License. http://opensource.org/licenses/MIT. [Consulta: 15 de Septiembre de 2014].

[21] Opera Software. Opera Web Browser. http://www.opera.com/es-ES/computer/windows. [Consulta: 17 de Septiembre de 2014].

[22] Project Management Institute, Inc. PMBOK Guide and Standards. http://www.pmi.org/PMBOK-Guide-and-Standards.aspx. [Consulta: 16 de Septiembre de 2014].

[23] HTML Purifier. Standards-Compilant HTML Filtering. http://htmlpurifier.org/. [Consulta: 21 de Septiembre de 2014].

[24] Tobias Ratschiller. PHP My Admin. http://www.phpmyadmin.net/home_page/index.php. [Consulta: 16 de Septiembre de 2014].

[25] Seguridad Social. Contingencias Seguridad Social. http://www.seg-social.es/Internet_1/Trabajadores/CotizacionRecaudaci10777/Regimenes/RegimenGeneraldelaS10957/InformacionGeneral/index.htm. [Consulta: 20 de Septiembre de 2014].

[26] Sublime HQ Pty Ltd. Sublime Text. http://www.sublimetext.com/. [Consulta: 16 de Septiembre de 2014].

[27] The Apache Software Foundation. Apache HTTP server. http://httpd.apache.org/. [Consulta: 16 de Septiembre de 2014].

[28] The PHP Group. PHP. http://php.net/. [Consulta: 16 de Septiembre de 2014].

[29] Trello, Inc. Trello. https://trello.com/. [Consulta: 15 de Septiembre de 2014].

[30] Trovit. Salario medio programador junior 2014. http://empleo.trovit.es/1543/salarios-programador-junior. [Consulta: 20 de Septiembre de 2014].

[31] Twitter Inc. Sobre twitter. https://about.twitter.com/es/company/. [Consulta: 15 de Septiembre de 2014].

[32] University of Regina. Crow's Foot Notation. http://www2.cs.uregina.ca/~bernatja/crowsfoot.html. [Consulta: 16 de Septiembre de 2014].

[33] World Wide Web Consortium (W3C). Cascading Style Sheets. http://www.w3.org/Style/CSS/specs. [Consulta: 15 de Septiembre de 2014].

[34] World Wide Web Consortium (W3C). Hypertext Markup Language. http://www.w3.org/TR/html5/. [Consulta: 15 de Septiembre de 2014].

[35] Julia Wester. What is Kanban? http://www.everydaykanban.com/what-is-kanban/. [Consulta: 20 de Septiembre de 2014].

[36] Yii Software LLC. Automatic code generator. http://www.yiiframework.com/doc/guide/1.1/en/topics.gii. [Consulta: 17 de Septiembre de 2014].

[37] Yii Software LLC. Yii framework. http://www.yiiframework.com/. [Consulta: 15 de Septiembre de 2014].

[38] Zurb Inc. Foundation version 4. http://foundation.zurb.com/. [Consulta: 15 de Septiembre de 2014].

Anexo A

Versión final de las interfaces gráficas

Figura A.1: Captura de pantalla de tableta. Acceso a la aplicación.

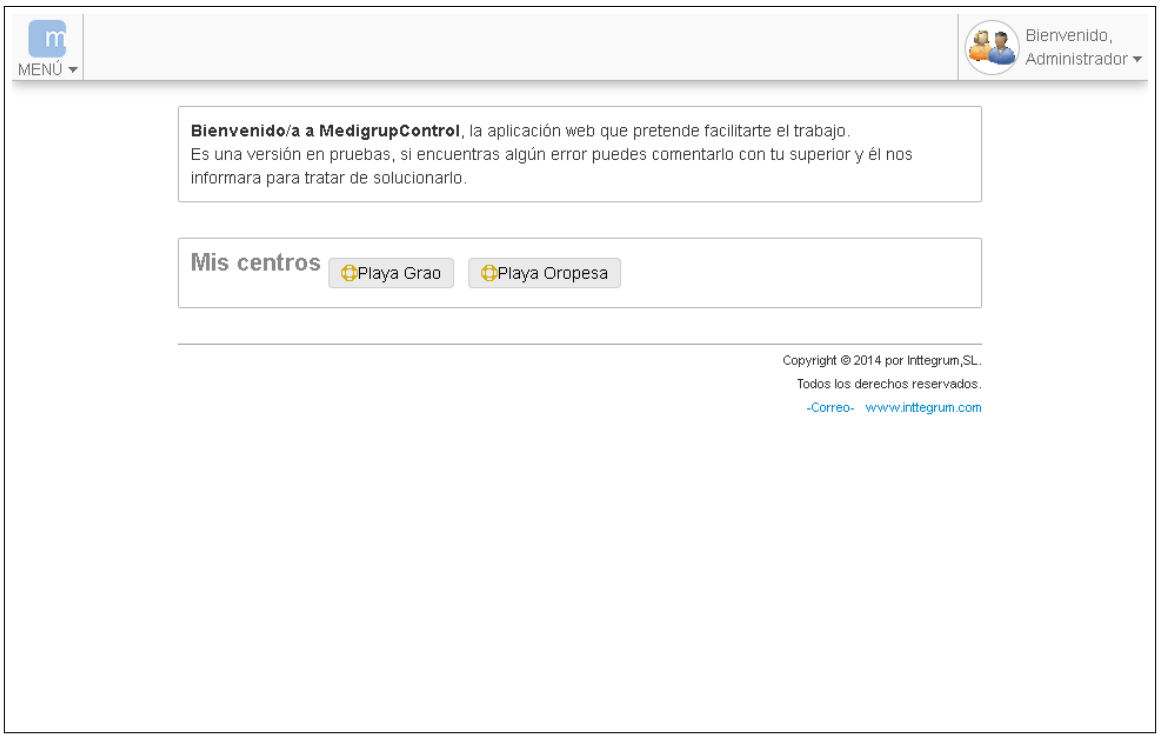

Figura A.2: Captura de pantalla de tableta. Pantalla de entrada.

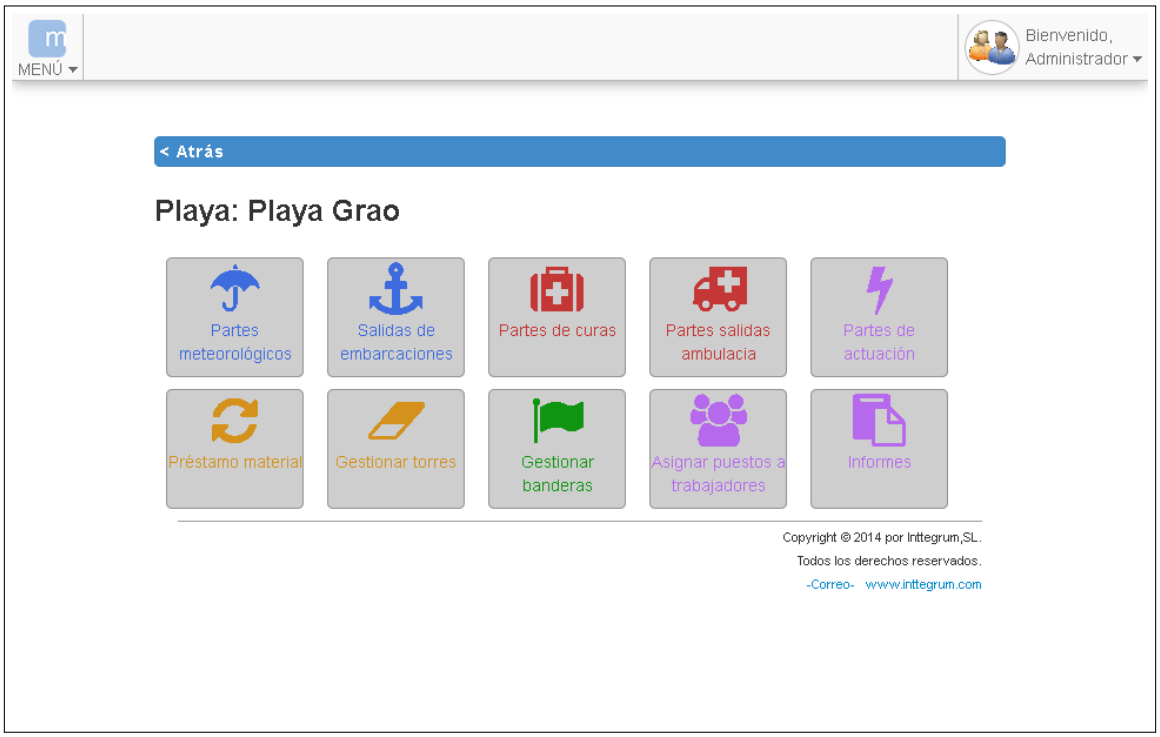

Figura A.3: Captura de pantalla de tableta. Menú principal.

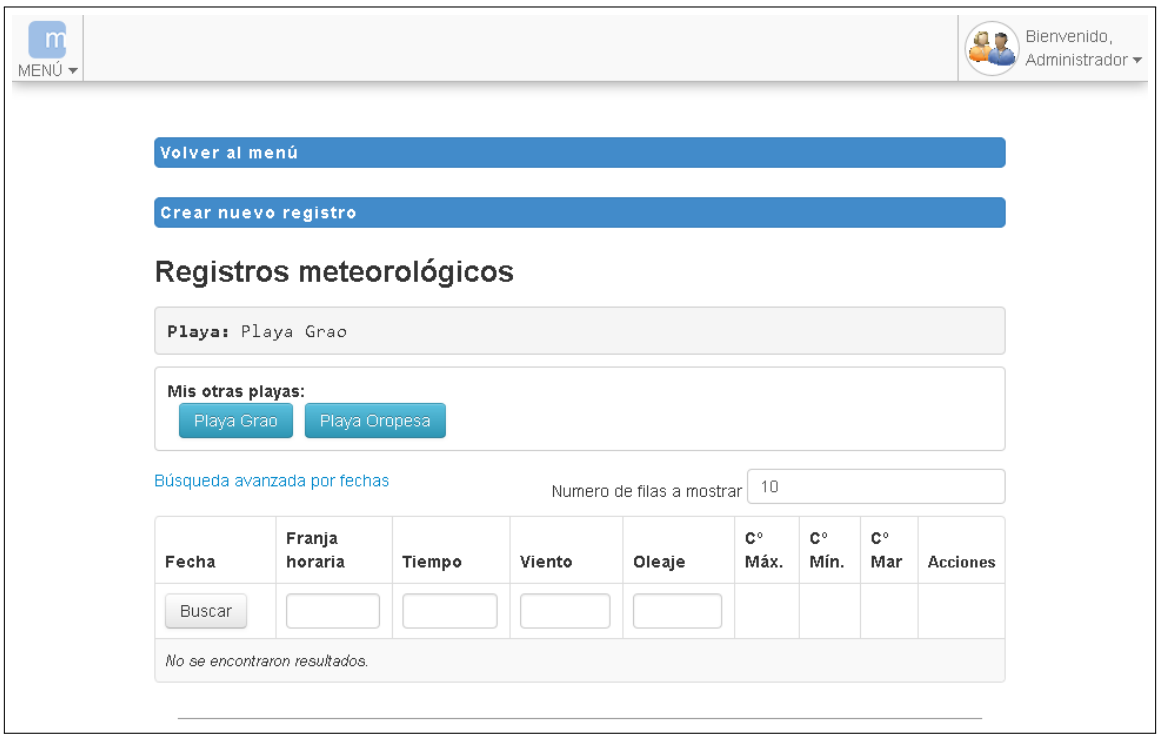

Figura A.4: Captura de pantalla de tableta. Gestión registro meteorológico.

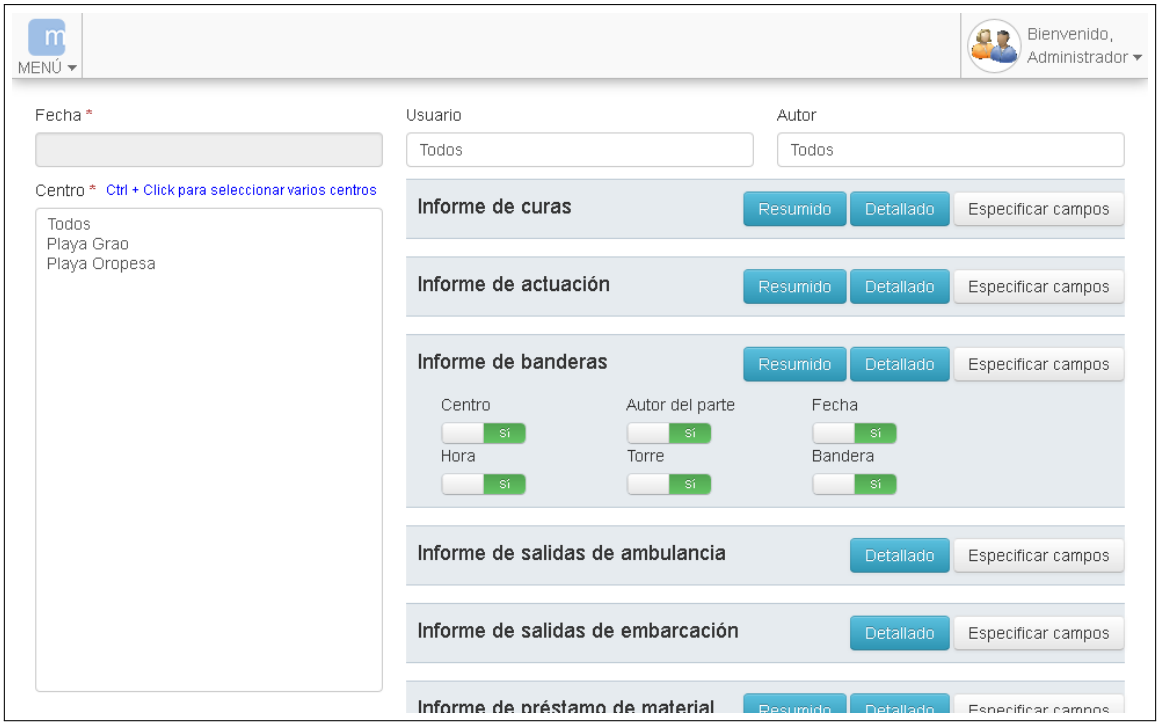

Figura A.5: Captura de pantalla de tableta. Generador de informes.

Anexo B

Manual de usuario

Gestión general de aplicación

Usuarios(Roles - Tipos)

Jefe de playa

Tipo: Playa.

Son los usuarios encargados de las playas.

Socorrista

Tipo: Playa.

Son los usuarios encargados del socorrismo en las playas.

DUE

Tipo: Playa.

Son enfermeros/as que están trabajando en las playas.

Patrón

Tipo: Playa.

Son patrones de embarcaciones que trabajan en las playas.

Baño asistido

Tipo: Playa.

Son los usuarios que trabajan en las playas y solo tienen acceso a los partes de préstamo de material.

Usuarios(Gestionarlos)

Consultar todos los usuarios

→ *MENU* → *ADMINISTRAR APLICACIÓN* → *USUARIOS*

Aquí pueden verse todos los usuarios de la aplicación. Es posible filtrar por jefe, por correo, por nivel, buscar por nombre por edad, ver en que centros trabaja...

Crear usuarios

→ *MENU* → *ADMINISTRAR APLICACIÓN* → *USUARIOS* → *NUEVO USUARIO*

Para la creación de un nuevo usuario hay que tener en cuenta que cada usuario tiene que ser una persona que exista físicamente, no una entidad ni un grupo de personas. Cada persona tiene su propio usuario.

El jefe directo es la persona a la que tiene que pedirle vacaciones y cualquier otra cosa.

Eliminar y editar usuarios

→ *MENU* → *ADMINISTRAR APLICACIÓN* → *USUARIOS* → *ELIMINAR*

Solo se permiten eliminar usuarios si los acabamos de crear. Si el usuario ya ha sido asignado a cualquier centro o ya ha creado algo en la aplicación, lo único que podremos hacer es editar sus datos y darlo de baja.

Cambiar contraseña

→ *MENU* → *ADMINISTRAR APLICACIÓN* → *USUARIOS* → *CAMBIAR CONTRASEÑA*

El propio usuario podrá cambiar la contraseña desde su menú de usuario (el de la parte superior a la derecha) pero si algún administrador quiere o necesita cambiar la contraseña de cualquier usuario también podrá realizar esta acción desde este menú.

Asignar usuarios a centros de trabajo

→ *MENU* → *ADMINISTRAR APLICACIÓN* → *USUARIOS* → *ASIGNAR CENTROS*

Dentro de esta sección podemos ver que centros tiene asignados un usuario, quitarle centros presionando el botón de eliminar y también añadirle un nuevo centro seleccionándolo en el desplegable y presionando el botón de “asignar”.

Gestión de playas

Partes meteorológicos

Ver los partes

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES METEOROLÓGICOS

En esta sección se pueden ver todos los partes meteorológicos que se han creado, también es posible buscar por fechas o por franjas horarias o por viento...

(se facilita la posibilidad de cambiar de playa de forma rápida en el menú superior donde unos botones nos permiten acceder a las distintas playas que tiene asignadas el usuario activo)

Crear nuevo parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES METEOROLÓGICOS → CREAR NUEVO REGISTRO

La fecha que aparece por comodidad es la actual, pero en cualquier momento se puede modificar haciendo clic encima.

En el formulario de creación del parte, en caso que la temperatura tenga un decimal, éste deberá estar separado por un punto y no por una coma, de lo contrario el sistema no lo aceptará.

Ver detalles de un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES METEOROLÓGICOS → DETALLES

Si hay que ver más información de la que nos muestra la tabla principal podemos abrir la una ventana nueva con los detalles de un parte en particular. Esto se puede hacer presionando el botón que tenemos en la parte derecha de la tabla.

Modificar un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES METEOROLÓGICOS → EDITAR

Para modificar un parte simplemente pulsaremos el botón que tenemos a la derecha de la tabla cuyo nombre es “Editar”, esta edición se verá reflejada de forma inmediata tras guardar los cambios y la única forma de volver a los datos anteriores es volver a editar el parte y reescribirlos de forma manual. Es decir, una vez modificado un parte la información vieja se destruye.

Eliminar un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES METEOROLÓGICOS → ELIMINAR

Cuando sea imprescindible borrar un parte, la opción está a la derecha de la tabla en un

botón rojo que tiene un dibujo de una papelera. Una vez borrado no hay forma de deshacer este borrado, por lo tanto mucho cuidado con borrar partes.

Partes de curas

Ver los partes → *MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES DE CURAS*

En esta sección se pueden ver todos los partes de curas que se han creado. También es posible buscar por fechas o por franjas horarias.

Crear nuevo parte

→ *MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES DE CURAS → CREAR NUEVO PARTE*

La fecha y la hora que aparecen por comodidad son las actuales, pero en cualquier momento se puede modificar haciendo clic encima.

Si se produce una salida de ambulancia hay que marcar la opción y rellenar el formulario que se despliega.

Ver detalles de un parte

→ *MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES DE CURAS → DETALLES*

Si hay que ver más información de la que nos muestra la tabla principal podemos abrir la una ventana nueva con los detalles de un parte en particular. esto lo hacemos presionando el botón que tenemos en la parte derecha de la tabla

Modificar un parte

→ *MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES DE CURAS → EDITAR*

Para modificar un parte simplemente pulsaremos el botón que tenemos a la derecha de la tabla cuyo nombre es “Editar”, esta edición se verá reflejada de forma inmediata tras guardar los cambios y la única forma de volver a los datos anteriores es volver a editar el parte y reescribirlos de forma manual, es decir, una vez modificado un parte la información vieja se destruye

Eliminar un parte

→ *MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES DE CURAS → ELIMINAR*

Cuando sea necesario e imprescindible borrar un parte la opción está a la derecha de la tabla en un botón rojo que tiene un dibujo de una papelera, una vez borrado no hay forma de deshacer este borrado, por lo tanto mucho cuidado con borrar partes

Partes de actuación

Ver los partes

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES DE ACTUACIÓN

En esta sección se pueden ver todos los partes de curas que se han creado, también es posible buscar por fechas o por franjas horarias...

Crear nuevo parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES DE ACTUACIÓN → CREAR NUEVO PARTE

La fecha y la hora que aparecen por comodidad son las actuales, pero en cualquier momento se puede modificar haciendo clic encima.

Si se produce una salida de ambulancia hay que marcar la opción y rellenar el formulario que se despliega.

Si se ha realizado una salida de embarcación deberá marcarse la opción Salida embarcación y rellenarse el formulario desplegado. Para añadir todos los tripulantes de la embarcación hay que seleccionarlos uno a uno en el campo Tripulación y pulsar Añadir a la lista.

Ver detalles de un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES DE ACTUACIÓN → DETALLES

Si hay que ver más información de la que nos muestra la tabla principal podemos abrir la una ventana nueva con los detalles de un parte en particular. esto lo hacemos presionando el botón que tenemos en la parte derecha de la tabla

Modificar un parte

Los partes de actuación no pueden editarse y, en caso de error, deberá borrarse y crear un nuevo.

Eliminar un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES DE ACTUACIÓN → ELIMINAR

Cuando sea necesario e imprescindible borrar un parte la opción está a la derecha de la tabla en un botón rojo que tiene un dibujo de una papelera, una vez borrado no hay forma de deshacer este borrado, por lo tanto mucho cuidado con borrar partes.

Partes de salidas de ambulancias

Ver los partes

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES SALIDAS AMBULANCIA

En esta sección podemos ver todos los partes de las salidas de las ambulancias, está dividido en tres pestañas, la primera "general" es donde se ven los partes de salidas de ambulancias que no están relacionados con ninguna cura o ninguna actuación. En la segunda pestaña "Partes actuación" podemos ver los partes de salida de ambulancia que están asociados a un parte de actuación, es decir, las salidas de ambulancia que se han registrado desde el formulario de creación de partes de actuación. y la tercera pestaña "Partes curas" nos permite ver los partes de salidas de ambulancia que están ligados a partes de curas.

Crear nuevo parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES SALIDAS AMBULANCIA → CREAR NUEVO PARTE

La fecha y la hora que aparecen por comodidad son las actuales, pero en cualquier momento se puede modificar haciendo clic encima.

Trabajador es la persona que se ha ido dentro de la ambulancia, normalmente será un DUE.

Ver detalles de un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES SALIDAS AMBULANCIA → DETALLES

Si hay que ver más información de la que nos muestra la tabla principal podemos abrir la una ventana nueva con los detalles de un parte en particular. esto lo hacemos presionando el botón que tenemos en la parte derecha de la tabla.

Modificar un parte

Solo pueden editarse los partes de salida de ambulancia que no están relacionados con un parte de cura o un parte de actuación.

Si por alguna razón hay que modificar un parte de salida de ambulancia relacionado con otro parte de curas o de actuación, hay que consultar la sección de como modificar un parte de curas o de actuación.

Eliminar un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PARTES SALIDAS AMBULANCIA → ELIMINAR

Solo pueden eliminarse los partes de salida de ambulancia que no están relacionados con un parte de cura o un parte de actuación.

Para borrar un parte relacionado, hay que borrar antes el parte de curas o de actuación con el cual está relacionado.

Partes de salidas de embarcación

Ver los partes

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → SALIDAS DE EMBARCACIONES

En esta sección podemos ver todos los partes de las salidas de embarcaciones. Está dividido en dos pestañas, la primera "general" es donde se ven los partes de salidas de embarcaciones que no están relacionados con ninguna cura o ninguna actuación. En la segunda pestaña "Partes actuación" podemos ver los partes de salida de embarcación que están asociados a un parte de actuación, es decir, las salidas de embarcación que se han registrado desde el formulario de creación de partes de actuación.

Crear nuevo parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → SALIDAS DE EMBARCACIONES → CREAR NUEVO PARTE

La fecha y la hora que aparecen por comodidad son las actuales, pero en cualquier momento se puede modificar haciendo click encima.

La tripulación se añade seleccionando un tripulante de la lista desplegable y pulsando el botón "Añadir a la lista". si hay que quitar algún tripulante de la lista pincharemos el botón rojo con una x que está situado a la izquierda de cada nombre.

Ver detalles de un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → SALIDAS DE EMBARCACIONES → DETALLES

Si hay que ver más información de la que nos muestra la tabla principal podemos abrir la una ventana nueva con los detalles de un parte en particular. esto lo hacemos presionando el botón que tenemos en la parte derecha de la tabla

Modificar un parte

Solo pueden editarse los partes de salidas de embarcaciones que no están relacionados con un parte de cura o un parte de actuación.

Si por alguna razón hay que modificar un parte de salida de embarcación relacionado con otro parte de curas o de actuación, hay que consultar la sección de como modificar un parte de curas o de actuación.

Eliminar un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → SALIDAS DE EMBARCACIONES → ELIMINAR

Solo pueden eliminarse los partes de salida de embarcación que no están relacionados con un parte de cura o un parte de actuación.

Para borrar un parte relacionado, hay que borrar antes del parte de curas o de actuación con el cual está relacionado.

Préstamo de material anfibio

Ver los partes

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PRÉSTAMO MATERIAL

En esta sección podemos ver todos los partes de préstamo de material.

Crear nuevo parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PRÉSTAMO MATERIAL → CREAR NUEVO PARTE

La fecha y la hora que aparecen por comodidad son las actuales, pero en cualquier momento se puede modificar haciendo clic encima.

En el formulario de creación deben introducirse el número de unidades que se han prestado en sus respectivos campos. Si no se ha prestado nada el campo deberá dejarse en 0, que es el valor por defecto. Si el material prestado no está en las opciones disponibles deberá añadirse en el campo Otros junto con el número de unidades prestadas.

Ver detalles de un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PRÉSTAMO MATERIAL → DETALLES

Si hay que ver más información de la que nos muestra la tabla principal podemos abrir la una ventana nueva con los detalles de un parte en particular. esto lo hacemos presionando el botón que tenemos en la parte derecha de la tabla.

Modificar un parte

Las modificaciones de un parte una vez guardado no pueden deshacerse a no ser que se vuelva a modificar el parte de forma manual.

Eliminar un parte

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → PRÉSTAMO MATERIAL → ELIMINAR

Un parte borrado no podrá recuperarse.

Gestionar torres

Ver las torres de una playa

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → GESTIONAR TORRES

Esta sección es solo para los jefes de playa o administradores de la aplicación, en ella se ven todas las torres que hay dadas de alta en una playa, las torres sirven para luego poder poner las banderas.

Crear una nueva torre

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → GESTIONAR TORRES → AÑADIR UNA TORRE

A la hora de añadir una torre solo hace falta darle un nombre, suelen ser números, por ejemplo: Torre 1, Torre 2.

Modificar una torre

Las modificaciones de una torre una vez guardada no pueden deshacerse.

Eliminar una torre

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → GESTIONAR TORRES → ELIMINAR

Una torre borrada no podrá recuperarse.

Parte de banderas

Ver el histórico de banderas

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → GESTIONAR BANDERAS

Esta sección permite visualizar las banderas de una playa ordenadas por fecha, estando en la parte superior las del día más próximo.

(se facilita la posibilidad de cambiar de playa de forma rápida en el menú superior donde unos botones nos permiten acceder a las distintas playas que tiene asignadas el usuario activo)

Crear nuevo parte de banderas

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → GESTIONAR BANDERAS → AÑADIR BANDERAS

Se pueden añadir banderas de forma sencilla presionando en el color deseado para cada torre, también podemos marcar todas las torres con el mismo color presionando en la bandera del color deseado que hay en la primera fila llamada "Todas".

(se facilita la posibilidad de cambiar de playa de forma rápida en el menú superior donde unos botones nos permiten acceder a las distintas playas que tiene asignadas el usuario activo)

Modificar un parte de banderas

No es posible modificar un parte de banderas, hay que eliminarlo y crearlo de nuevo.

Eliminar un parte de banderas

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → GESTIONAR BANDERAS → ELIMINAR

Para borrar un parte de banderas se apretara el botón de eliminar y una vez eliminado no podrá deshacerse, siendo la única opción para recuperarlo crearlo de nuevo.

Informes

Creación de informes

→ MENU → GESTIONAR PLAYAS → ACCEDER A MIS PLAYAS → INFORMES

Los informes solo están disponibles para los administradores de la aplicación.

Para la creación de informes deberá introducirse el rango de fechas deseado, el centro, el usuario y el autor.

El centro es la playa sobre la que se quiera hacer el informe. En caso de querer un informe de todas las playas deberá seleccionarse la opción Todos.

El usuario es el trabajador. En caso de querer un informe con las acciones de un trabajador en concreto deberá seleccionarse aquí. De lo contrario se debe seleccionar la opción "Todos".

El autor es el trabajador que ha creado el parte en la aplicación. Si no se quiere mostrar alguno en concreto deberá seleccionarse la opción Todos.

Para elegir los campos que se mostrarán en el informe se debe pulsar el botón Especificar campos y marcar como Sí los campos deseados.

Finalmente se debe pulsar el botón Generar informe para descargarlo en formato Excel (.xls.)